NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

THESIS

FLOW FIELD SURVEYS IN A TRANSONIC COMPRESSOR PRIOR TO INLET STEAM INGESTION TESTS

by

Ivan Villescas

September 2005

Thesis Advisor:	Garth Hobson
Second Reader:	Anthony Gannon

Approved for public release; distribution is unlimited

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE	*Form Approved OMB No. 0704-0188*

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington DC 20503.

1. AGENCY USE ONLY *(Leave blank)*	**2. REPORT DATE** September 2005	**3. REPORT TYPE AND DATES COVERED** Master's Thesis

4. TITLE AND SUBTITLE: Title (Mix case letters) Flow Field Surveys in a Transonic Compressor Prior to Inlet Steam Ingestion Tests	**5. FUNDING NUMBERS**
6. AUTHOR(S)	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943-5000	**8. PERFORMING ORGANIZATION REPORT NUMBER**
9. SPONSORING /MONITORING AGENCY NAME(S) AND ADDRESS(ES) Naval Air Systems Command (Air 4.4) Naval Air Warfare Center-Aircraft Div Propulsion and Power Engineering Patuxent River, Maryland	**10. SPONSORING/MONITORING AGENCY REPORT NUMBER**

11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release; distribution is unlimited.	**12b. DISTRIBUTION CODE**

13. ABSTRACT (maximum 200 words)

Investigating the effect of steam ingestion into an aircraft jet engine is necessary to improve understanding of stall and surge in transonic axial compressors. Specifically, to understand the "pop stall" phenomenon experienced by naval fighter jet aircraft during steam catapult launches. Steam leakage from an aircraft carrier catapult system can be ingested into the intake and cause stall or surge in a jet engine upon takeoff. It is important to understand the conditions under which this occurs as the Navy prepares for the fielding of the single engine F-35C, the aircraft carrier variant of the Joint Strike Fighter. This project prepares the structure and instrumentation to investigate the inlet distortion and effects of steam ingestion on a transonic axial compressor. A compressor test facility, including mechanical equipment, data acquisition system, and remote digital control system, was configured to test a transonic compressor rotor, similar to what will be used in the Joint Strike Fighter. Rotor inlet and exit velocity profiles were measured with a three-hole probe to obtain a set of baseline data before future experiments.

14. SUBJECT TERMS Compressor, Transonic, Steam Ingestion, Inlet Distortion, Turbomachinery, Pop Stall, Rotor	**15. NUMBER OF PAGES** 159
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT Unclassified	**18. SECURITY CLASSIFICATION OF THIS PAGE** Unclassified	**19. SECURITY CLASSIFICATION OF ABSTRACT** Unclassified	**20. LIMITATION OF ABSTRACT** UL

THIS PAGE INTENTIONALLY LEFT BLANK

FLOW FIELD SURVEYS IN A TRANSONIC COMPRESSOR PRIOR TO INLET STEAM INGESTION TESTS

Ivan J. Villescas
Lieutenant, United States Navy
B.S., Rennselaer Polytechnic University, 1996

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN MECHANICAL ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL
September 2005

Author: Ivan Villescas

Approved by: Garth Hobson
Thesis Advisor

Anthony Gannon
Co-Advisor

Anthony Healey
Chairman, Department of Mechanical and Astronautical Engineering

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

Investigating the effect of steam ingestion into an aircraft jet engine is necessary to improve understanding of stall and surge in transonic axial compressors. Specifically, to understand the “pop stall” phenomenon experienced by naval fighter jet aircraft during steam catapult launches. Steam leakage from an aircraft carrier catapult system can be ingested into the intake and cause stall or surge in a jet engine upon takeoff. It is important to understand the conditions under which this occurs as the Navy prepares for the fielding of the single engine F-35C, the aircraft carrier variant of the Joint Strike Fighter. This project prepares the structure and instrumentation to investigate the inlet distortion and effects of steam ingestion on a transonic axial compressor. A compressor test facility, including mechanical equipment, data acquisition system, and remote digital control system, was configured to test a transonic compressor rotor, similar to what will be used in the Joint Strike Fighter. Rotor inlet and exit velocity profiles were measured with a three-hole probe to obtain a set of baseline data before future experiments.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

THIS PAGE INTENTIONALLY LEFT BLANK

ACKNOWLEDGMENTS

Producing a thesis at the Naval Postgraduate School Turbopropulsion Laboratory was a unique experience. The TPL is a rare nexus of industry, academia, and the military.

Garth Hobson and Ray Shreeve, thank you for the guidance, looking over my shoulder and keeping me on track. Anthony Gannon, thanks for the great ideas and occupying the chair on my left, ready with answers to all my questions. John Gibson and Rick Still, thanks for redirecting me with common sense, when necessary.

The atmosphere at the TPL was perfect for learning and working and many thanks to all students and faculty who work there for making my thesis a pleasant and rewarding opportunity.

THIS PAGE INTENTIONALLY LEFT BLANK

I. INTRODUCTION

The F-35C is the Navy variant of the Joint Strike Fighter (JSF) and will utilize the F-135 engine, which uses a transonic compressor. Experiments conducted with catapult launches of an F-18, at Lakehurst Naval Air Engineering Station, shown in Figures 1 and 2, demonstrated that current jet engines are susceptible to catastrophic "pop stall" when exposed to excessive amounts of steam. The Joint Strike Fighter will be the first single engine jet fighter in recent years to be assigned to aircraft carrier based operations. The single engine nature of the JSF increases the severity of a potentially catastrophic "pop stall," particularly if it were to be exposed to large amounts of steam leakage during a catapult launch aboard an aircraft carrier at sea. (Donelson, 2003)

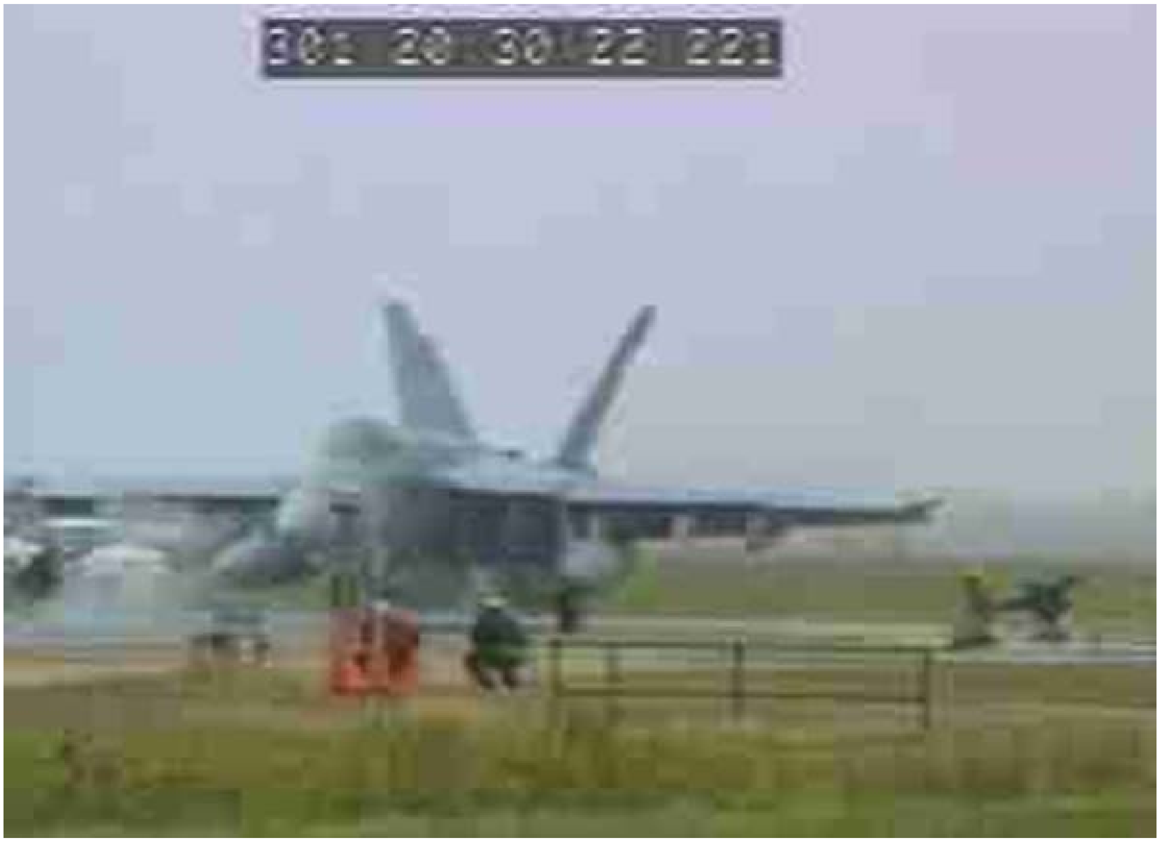


Figure 1. Lakehurst steam ingestion experiment. F-18 approaching steam leak during catapult launch.

The present goal of the Turbopropulsion Laboratory (TPL) at the Naval Post Graduate School (NPS) is to evaluate a small-scale transonic compressor during stall and surge when exposed to steam. The present study documents the equipment preparation and instrumentation design to conduct the evaluation, as well as baseline data obtained.

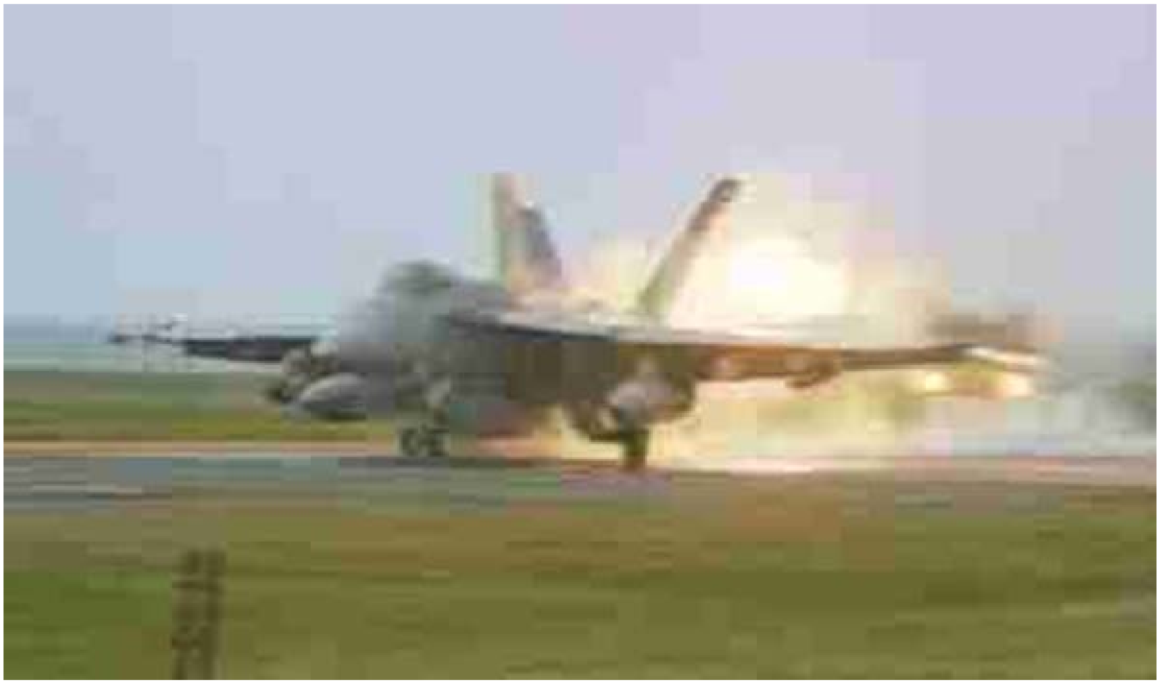


Figure 2. Lakehurst steam ingestion experiment. F-18 “Pop Stall” caused by steam leak during catapult launch.

To evaluate the “pop stall” phenomenon, a rotor similar to that used in military fighter jets was necessary. The rotor of the first stage of a transonic compressor was evaluated in the present study. The Sanger Rotor is a low aspect ratio transonic compressor. Nelson Sanger designed the compressor stage in 1996 at the NASA Lewis Research Center. The rotor was designed using CFD techniques and provided to the Naval Post Graduate School for experimental testing. (Sanger, 1996)

Pressures both upstream and downstream of the single rotor were evaluated by traversing a three-hole pressure probe at various locations. Prior to conducting stall experiments, a three-hole pressure probe was calibrated and a full baseline survey was completed. Data was obtained at 70, 80, 90, and 100 percent speed with mass flow rate varied from choke, peak efficiency, and to stall.

This project is the beginning of an in depth study of stall and surge in transonic compressors. The current project prepares the instrumentation and baselines the performance of a transonic rotor. Future projects will include the injection of steam upstream of the rotor. With the baseline completed, the calculation of stall margin and its change as steam is added to the system can be conducted.

.

THIS PAGE INTENTIONALLY LEFT BLANK

II. EXPERIMENTAL FACILITY AND EQUIPMENT

The current research was conducted at the Turbopropulsion Laboratory within the Department of Mechanical and Astronautical Engineering of the Naval Postgraduate School.

A. TRANSONIC COMPRESSOR RIG

The Transonic Compressor Rig (TCR) was originally built to test a prototype transonic compressor stage and was designed by Professor Michael Vavra. The configuration tested is shown in Figure 3 and was detailed extensively by O'Brien and Papamarkos. (O'Brien, 2000; Papamarkos, 2004) The TCR was set up to evaluate the Sanger transonic rotor. The test compressor was driven by two opposed air turbine stages, which were supplied with air by an Allison Chalmers axial compressor, and was capable of testing axial compressors up to 30,000 RPM.

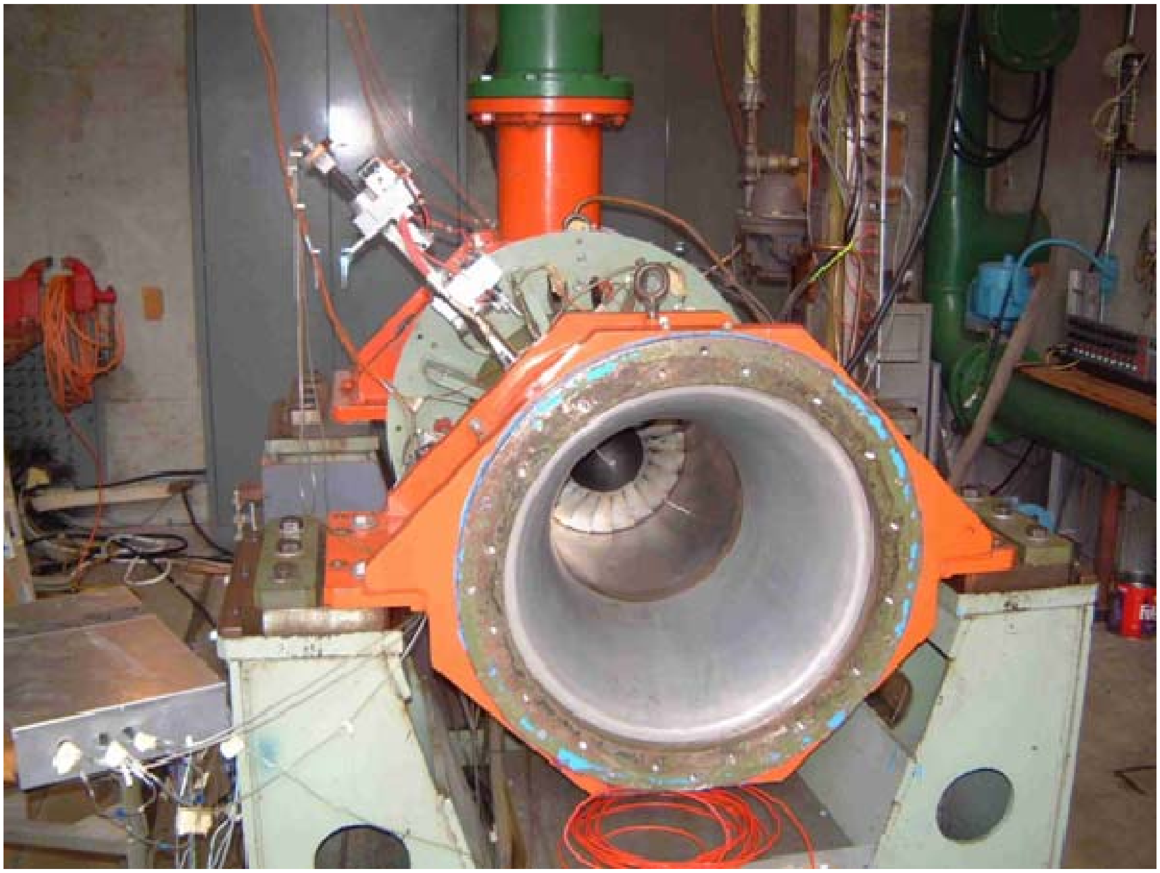


Figure 3. Transonic compressor rig in test cell with inlet piping removed.

Sanger, at the NASA Lewis Research Center, designed the compressor rotor as part of a complete fan stage. It was a completely Computational Fluid Dynamic (CFD) designed stage. Details of the design and design considerations are contained in TPL Technical Note 99-01. (Sanger, 1999) The stage was manufactured by numerical machining, specifically for testing and evaluation at the NPS Turbopropulsion Laboratory. Several experimental studies have been conducted on the stage. (Roberts, 2003; Gannon, Hobson, Shreeve, 2004) The rotor profile is shown on the TCR in Figure 4.

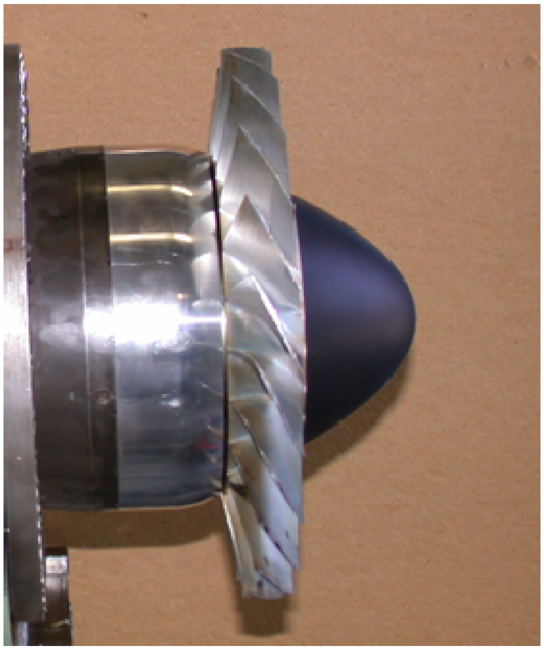


Figure 4. View of the rotor-only configuration tested.

The rotor was manufactured from a high strength aluminum alloy (7075-T6) and consisted of 22 blades. In this experiment, the rotor was tested with a parabolic spinner, which replaced the conical spinner used by O'Brien. (O'Brien, 2000; Gannon et al., 2004)

In the majority of previous studies, the entire stage was evaluated to verify the accuracy and validity of CFD design. For the current experiment, the rotor was evaluated without the stator and data was obtained for the rotor only, which was simpler to test mechanically and the future steam ingestion tests would quantify the effect on the rotor-only. In addition, the honeycomb used to straighten the flow downstream of the stage was removed. Design parameters for the rotor are reproduced from Sanger's "Design Methodology" in Table 1 below. (Sanger, 1996)

Table 1. Sanger rotor design parameters.

Parameter	Value
Pressure Ratio	1.61
Tip Speed	33.02 m/s (1300ft/s)
Design Speed	27085 rpm
Design Mass Flow	7.75 kg/s (17.05lb/s)
Specific Mass Flow	170.88 kg/s-m^2 (35lbm/s-ft^2)
Specific Head Rise	.246
Tip Inlet Relative Mach Number	1.28
Aspect Ratio	1.2
Hub/Tip Radius Ratio	.51
Rotor Inlet Ramp Angle	28.2
Number of Rotor Blades	22
Tip Solidity (Rotor)	1.3
Outside Diameter	0.2794m (11 inches)
Rotor Diffusion Factor – Tip	.4
Rotor Diffusion Factor - Hub	.47

A compressor map from "A Transonic Compressor State Part 1," (Gannon, et al., 2004) for the rotor only is shown in Figures 5 and 6. The stage design point is highlighted for the 100% speed line. The CFD data on the compressor map was obtained using SWIFT, a multi-block computer code created for analysis of three-dimensional viscous flows in turbomachinery. (Chima, 1998) A complete stage compressor map, including rotor and stator, can be found in Appendix B.

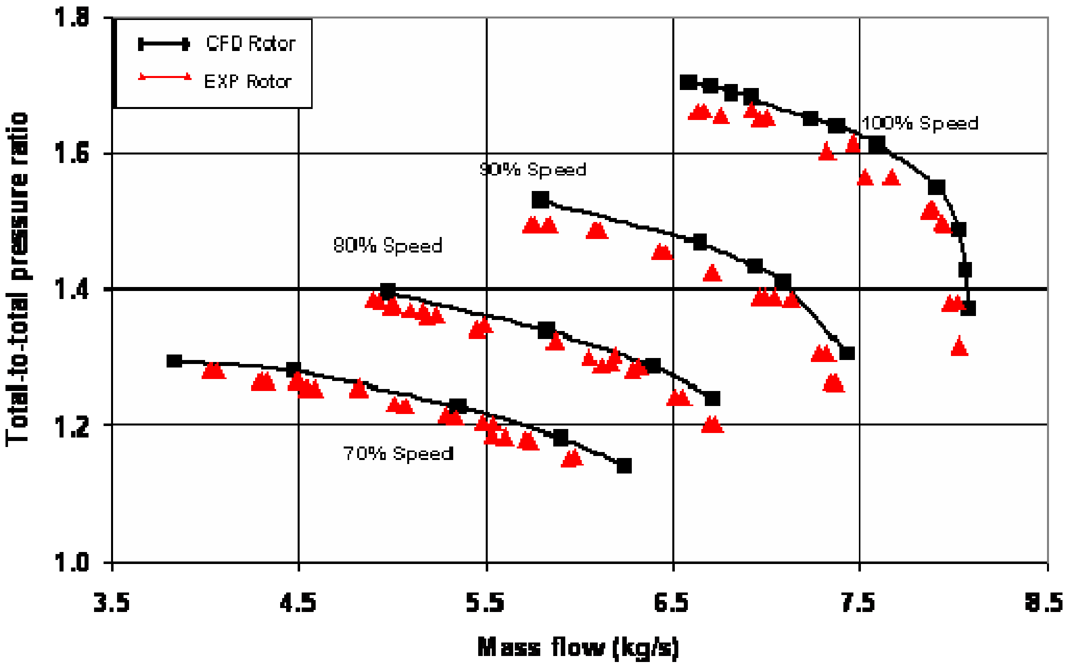


Figure 5. Total pressure ratio compressor map.

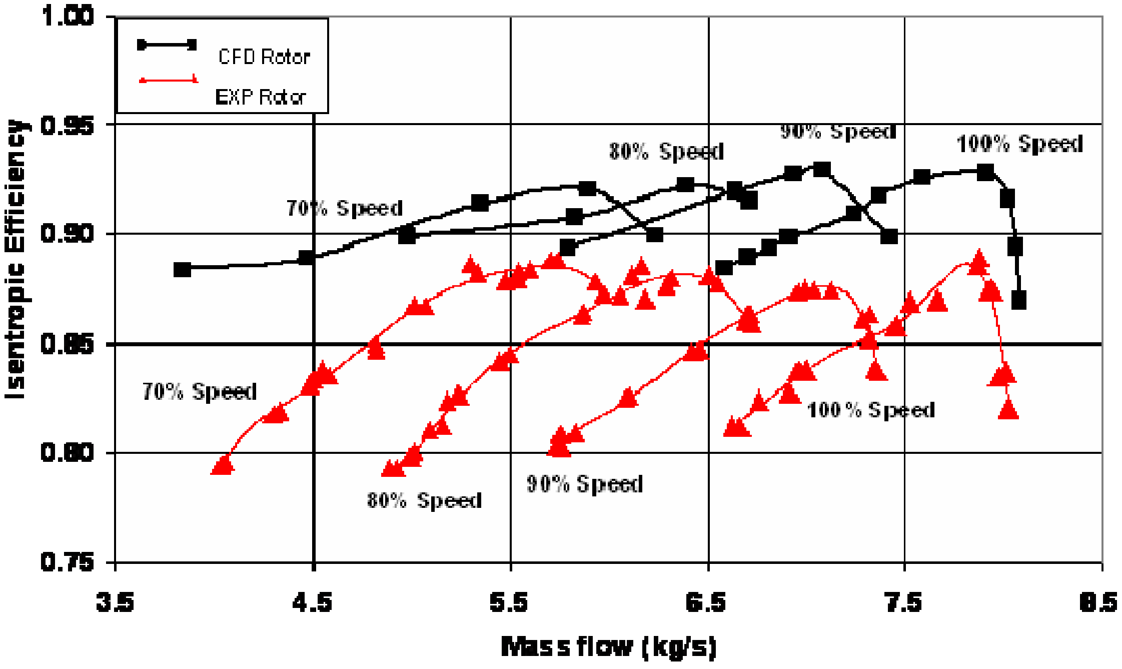


Figure 6. Isentropic efficiency compressor map.

A steel case wall was used to mount instrumentation for measurements. The four holes labeled in Figure 7 were designed for mounting a probe actuator for flow measurements. The gold band represents the location of the rotor, with holes one and two located upstream of the rotor and holes three and four downstream. Additional details of the case wall can be found in Appendix A.

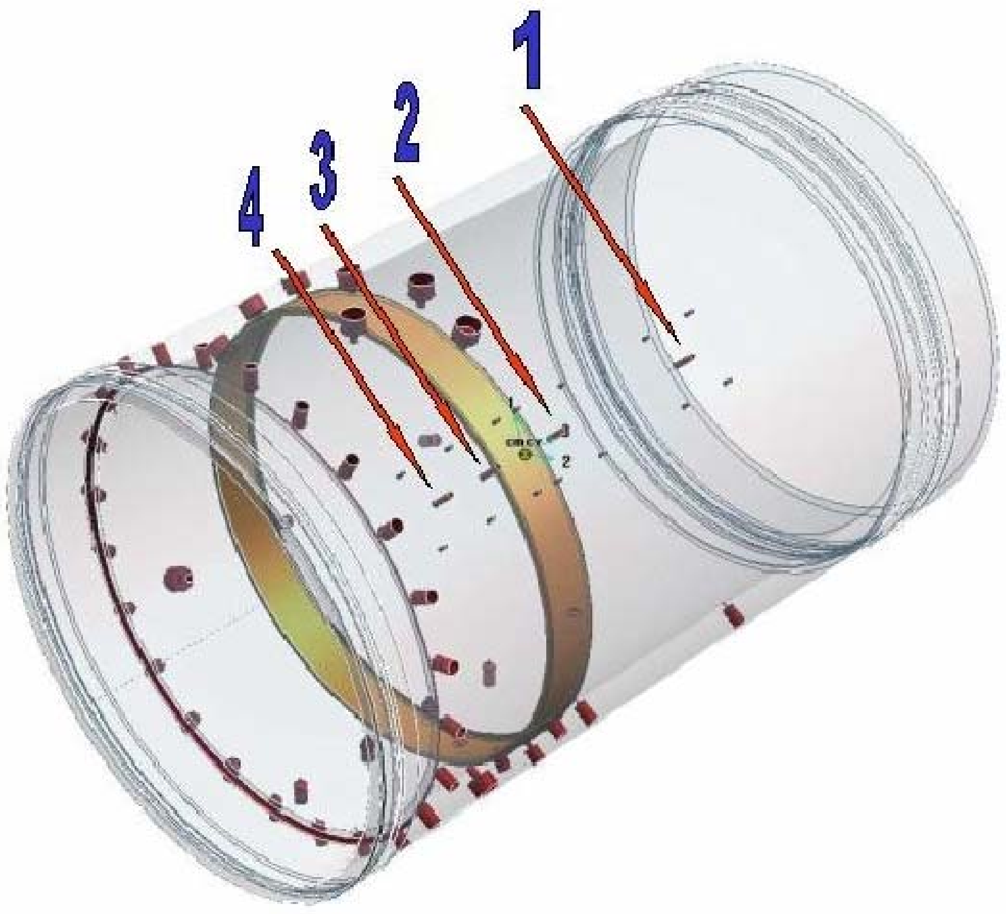


Figure 7. Compressor case wall.

Hole locations one and four were used in the current project. Hole one was located approximately 11.15cm (4.39 inches) upstream of the leading edge of the rotor, while hole four was located approximately 5.94cm (2.34 inches) downstream of the trailing edge of the rotor. Hole one was also located axially forward of the nose cone, giving access to a complete radius of inlet flow sampling, from center to case wall.

In the TCR, air was drawn from the atmosphere through an inlet throttle valve and into a settling chamber, labeled "1" in Figure 8. A 46cm diameter, five-meter long inlet pipe connected the chamber to the test compressor. The inlet pipe also contained a nozzle, labeled "2" in Figure 8, for flow rate measurement, and a transition duct to the 27.94cm diameter case wall. (Papamarkos, 2004)

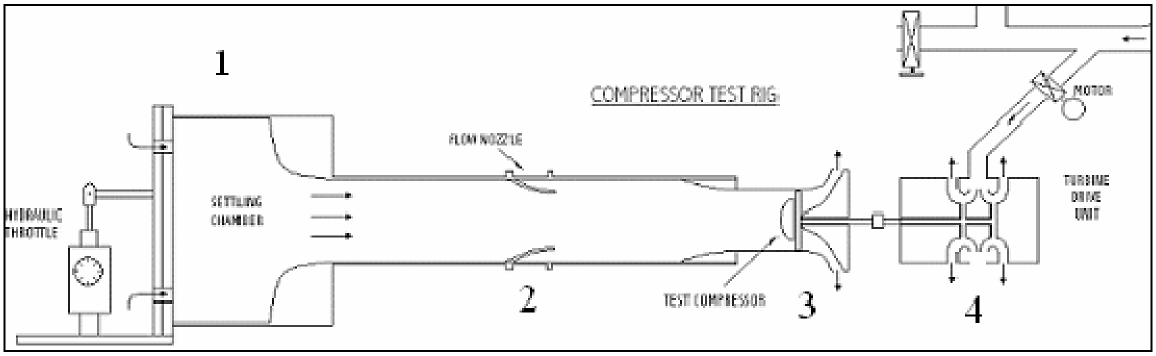


Figure 8. Compressor test rig air flow.

B. STEAM/INLET DISTORTION GENERATION

A Sussman model SVS600 steam generator was installed in the test cell adjacent to the TCR to provide steam for the compressor inlet. The SVS600 contained a 612kW boiler and could provide steam at a Boiler Horsepower (BHP) rating of 62.4. This corresponded to supply steam of 82.1 kgs (1806 lbs) per hour at 100C (212F). The design pressure of the boiler is 1.034MPa (150psig). (Sussman, 2004) The steam supply system is shown in Figure 9.

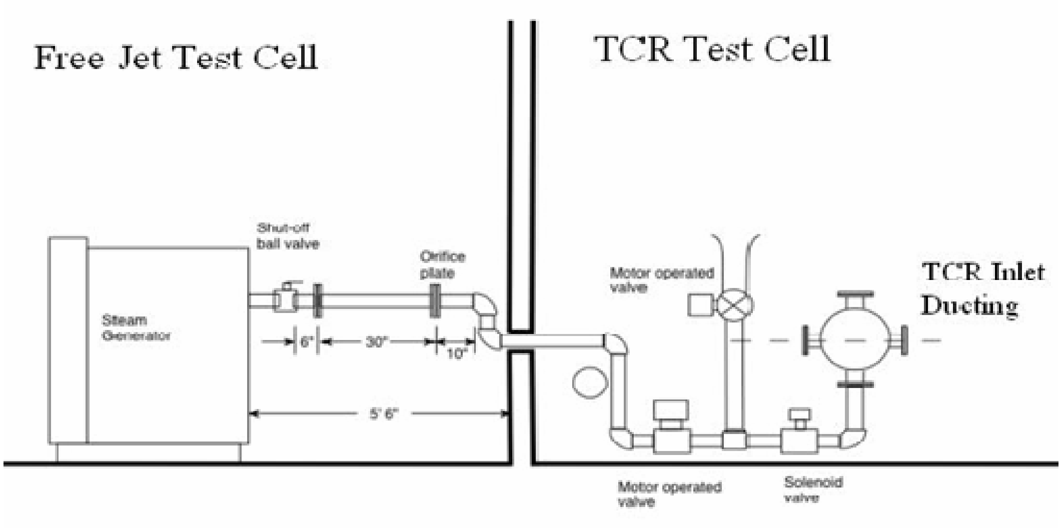


Figure 9. TCR Steam generation system.

The steam generated by the SVS600 will be injected upstream of the test compressor. Steam from the boiler will be piped through a manual shutoff valve and an orifice plate before entering the TCR test cell. Inside the test cell, steam will pass through a motor operated throttle control valve before mixing with atmospheric air, also controlled with a motor operated throttle control valve. A fast acting solenoid valve provides the capability of intermittently injecting steam into the compressor inlet. Figure 10 shows the interface of steam ingestion system with the TCR air inlet. The inlet distortion duct drawing in Appendix C outlines the steam injection piping connection to the compressor intake system.

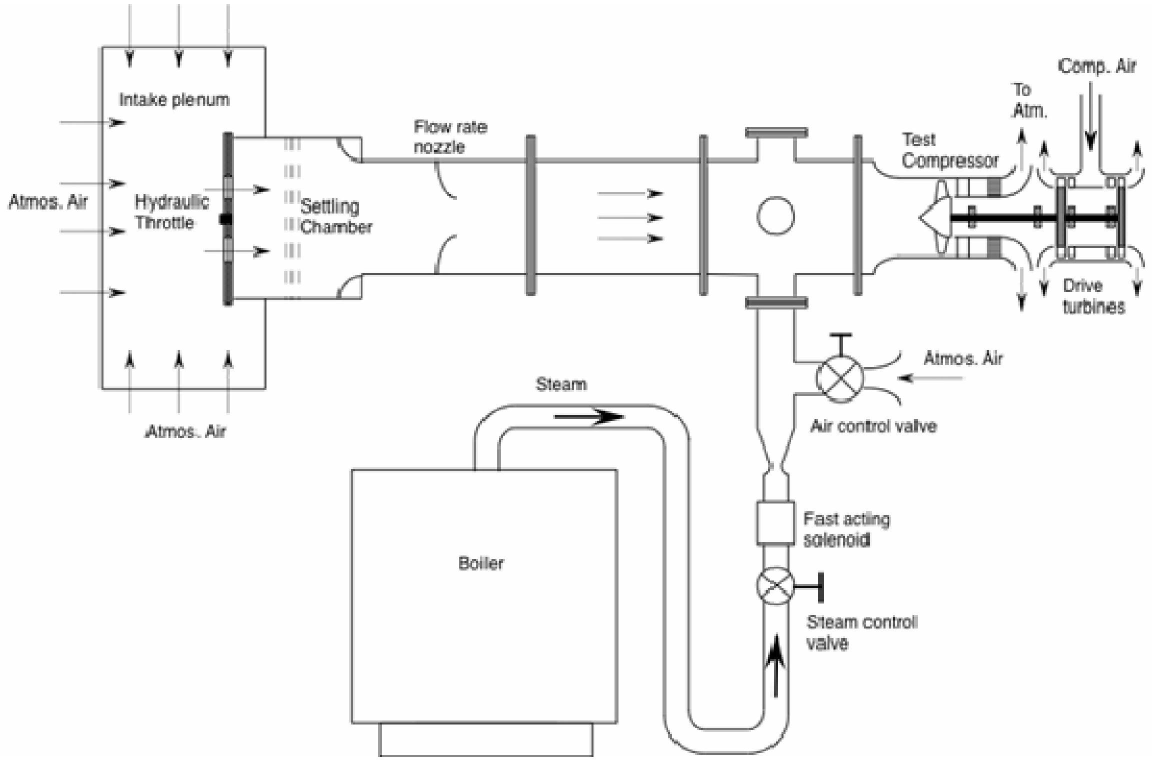


Figure 10. Steam ingestion system.

C. FREE JET TEST FACILITY

The free jet test facility was used to calibrate the three-hole probe. A mounting bracket provided positions for securing probes and also contained a sliding bracket capable of pitch movement. The free jet facility used air from the Allison Chalmers compressor to create a steady flow through a 10.8cm (4.25inch) nozzle. For Mach numbers above 0.3, a dump valve on the compressor was throttled to adjust the flow. For Mach numbers below 0.3, a manual shutoff valve located in the free jet cell was activated. The free jet is shown with the probe actuator mounted on the mounting bracket in Figure 11.

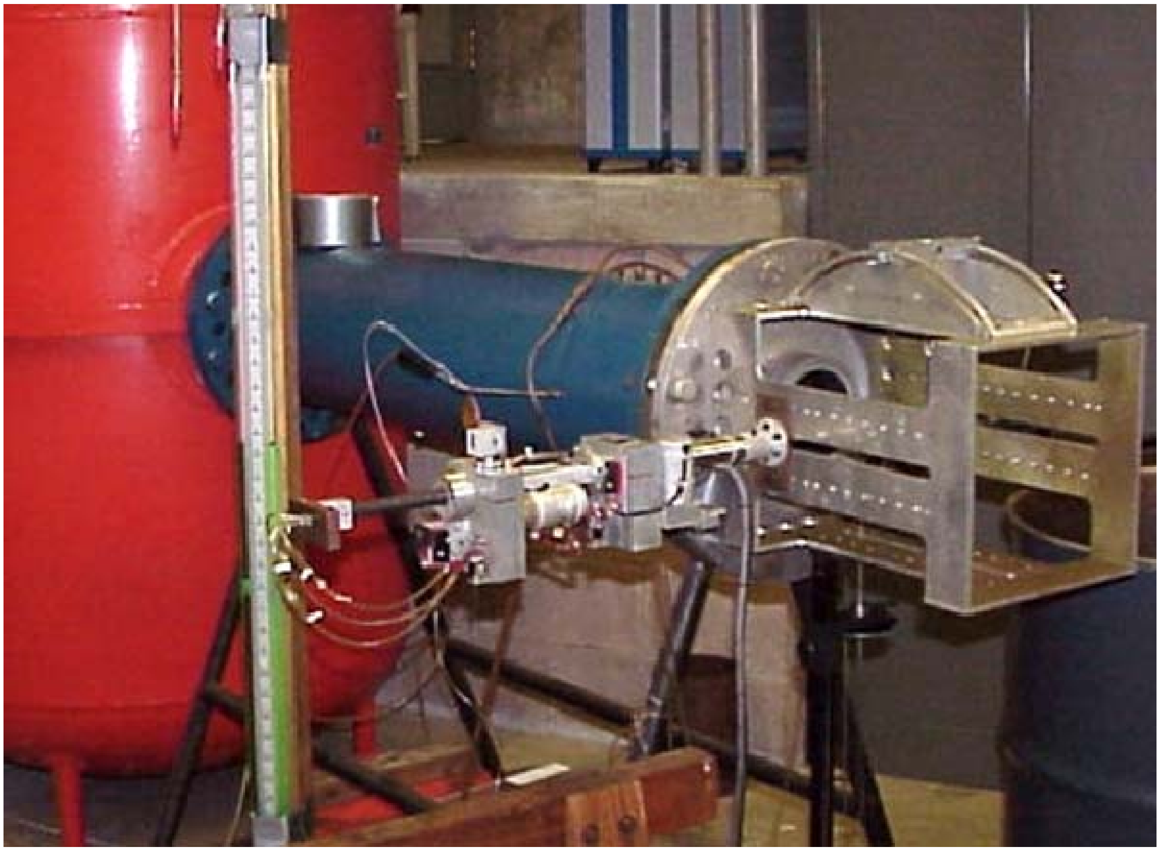


Figure 11. Three-hole probe within the probe actuator in the free jet assembly.

III. INSTRUMENTATION

A. PROBES

The primary probe used in this experiment was the three-hole cobra probe shown below in Figure 12. In addition, two additional probes were used in both the calibration and experimentation phases of this project. A kiel probe was used to detect the stagnation pressure and a pitot-static probe was used to detect the static pressure within the free jet. The probes positioned for calibration in the free jet test facility are show in Figure 13.

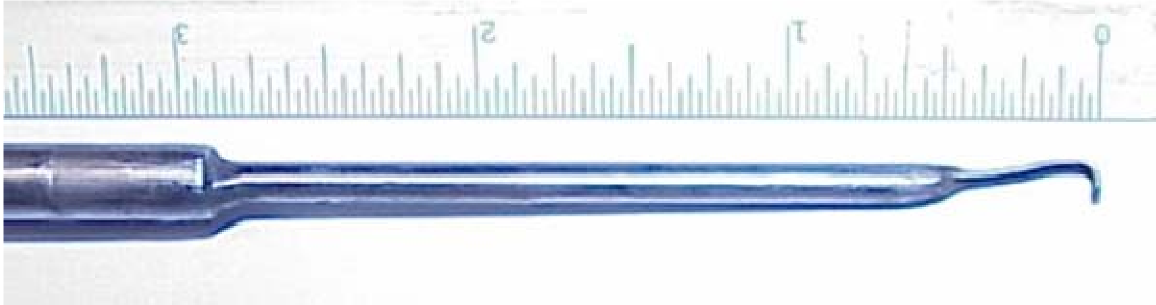


Figure 12. Three-hole cobra probe.

B. L.C. SMITH PROBE ACTUATOR

The L.C. Smith probe actuator, shown in Figure 11, was used to traverse and rotate the probe for calibration in the free jet and testing in the TCR. The probe actuator was initially set up in the same manner as described by Greco; however, the actuator was configured to an automatic PC controlled system. (Greco, 1995)

C. HPVEE/SCANIVALVE SYSTEM

The first calibration process used the Hewlett Packard Visual Engineering Environment (HPVEE) and Scanivalve system to obtain pressures in the cobra probe. The process used a setup identical to that outlined by Grossman, and screenshots of the post processing calculation program used in the current project are shown in Appendix D. (Grossman, 1997)

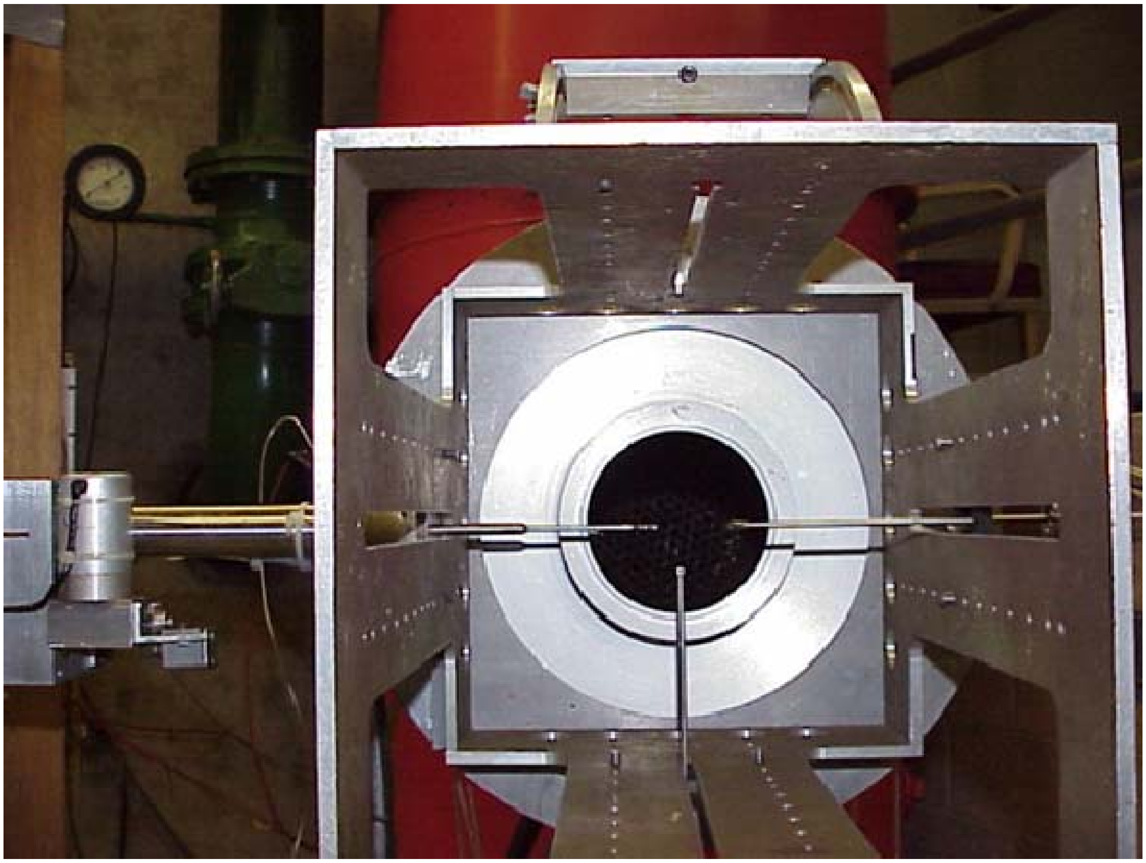


Figure 13. Probes positioned on the mounting bracket for calibration procedure on the free-jet test assembly.

D. OMEGA PX-138 PRESSURE TRANSDUCERS

The Omega PX-138 transducers used in the latter part of the experiment were miniature pressure transducers, one of which is shown in Figure 14. The pressure transducer used silicon pressure sensors, which were temperature-compensated. The PX138-015D5V model was capable of a differential pressure range of -15PSI to +15PSI with a linearity and hysteresis error of ±0.1% typical, ±0.5% max. The PX-138 was given a regulated 8VDC power supply and provided an output of 2.5VDC to 4.5VDC for the range of pressures applied in the experiment. (Omega, 2005)

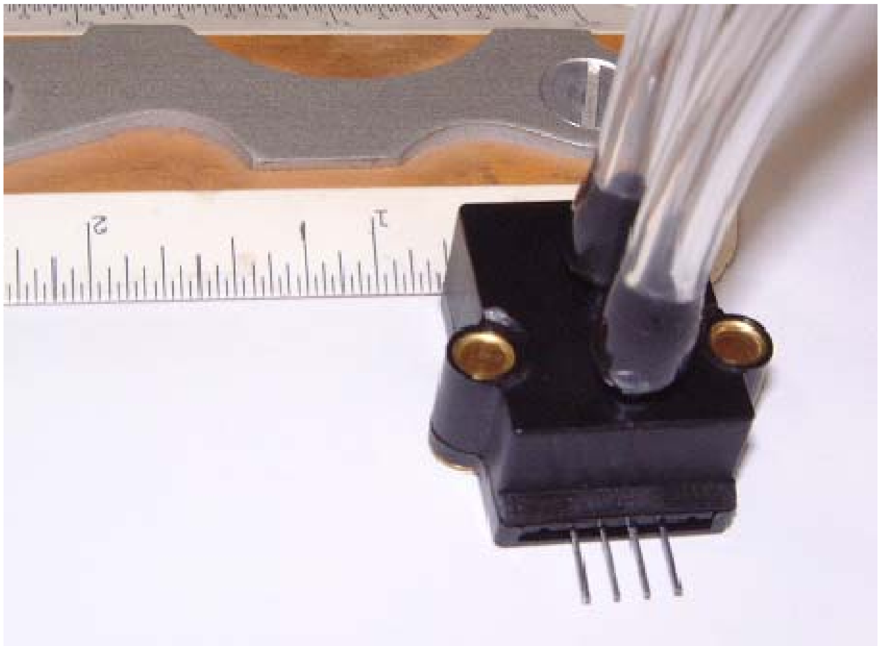


Figure 14. PX-138 pressure transducer

The pressure transducers remained calibrated throughout the experiment. Their pressure to voltage ratio was nearly linear and all five transducers would generally read atmospheric pressure within 10 Pascals, a less than 0.1% error. However, the PX-138 transducer had very small pinouts and came with a CX-136-4 plastic mating connector. Maintaining electrical connectivity was occasionally a problem due to the fragile nature of the pinouts and connectors.

E. PMD 1608FS DATA ACQUISITION DEVICE

The Personal Measurement Device™ (PMD) 1608FS Data Acquisition Device was a Universal Serial Bus (USB) 2.0 (full-speed) compatible, analog to digital (A/D) converter. The PMD 1608FS had a dedicated A/D converter for each channel and was capable of the simultaneous sampling of up to eight channels. Each channel was capable of sampling at 50 kS/s with a maximum of 200 kS/s for all channels. The PMD 1608FS was powered through the 5-volt USB port and it was controlled with a PC through the USB port with Matlab programming. A detailed block diagram from the user's manual is shown in Figure 15.

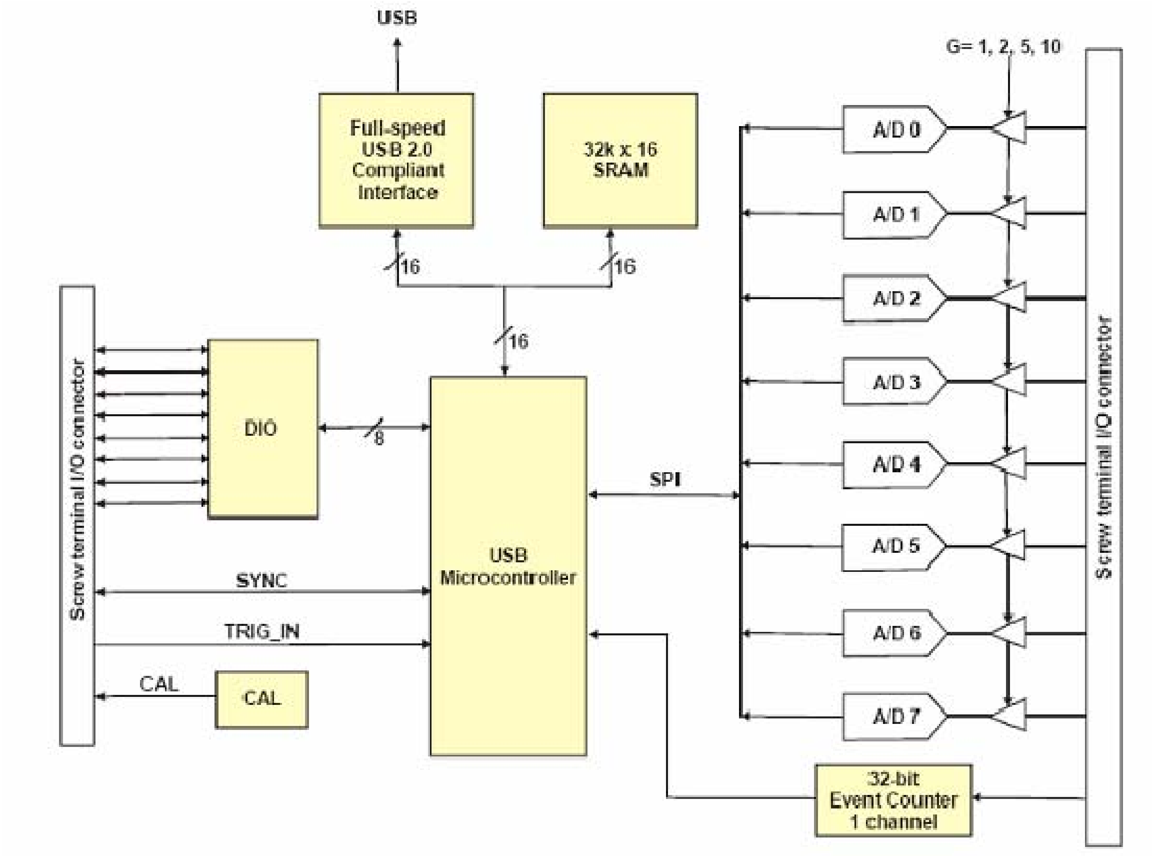

USB
Full-speed USB 2.0 Compliant Interface
32k x 16 SRAM
16
16
16
G= 1, 2, 5, 10
A/D 0
A/D 1
A/D 2
A/D 3
A/D 4
A/D 5
A/D 6
A/D 7
Screw terminal I/O connector
DIO
8
SPI
USB Microcontroller
SYNC
TRIG_IN
CAL
CAL
Screw terminal I/O connector
32-bit Event Counter 1 channel

Figure 15. PMD 1608FS A/D Module block diagram. (from Measurement Computing "PMD 1608FS", 2005)

F. USB ERB-24 REMOTE RELAY CONTROLLER

A USB ERB-24 Remote Relay Controller was used to control the traverse of the L.C. Smith gear. It was capable of being addressed by the PC via the USB 2.0 port. The USB ERB-24 was an electromechanical relay module with 24 electromechanical single pole double throw relays. The module was configured in two banks of eight and two banks of four relays. Each relay was capable of switching 6 amps at 28VDC with a minimum close time of 10 milliseconds and a minimum open time of 5 milliseconds. The module used a 9VDC power supply and was also controlled through the USB port with Matlab programming. A detailed block diagram from the user's manual is shown in Figure 16.

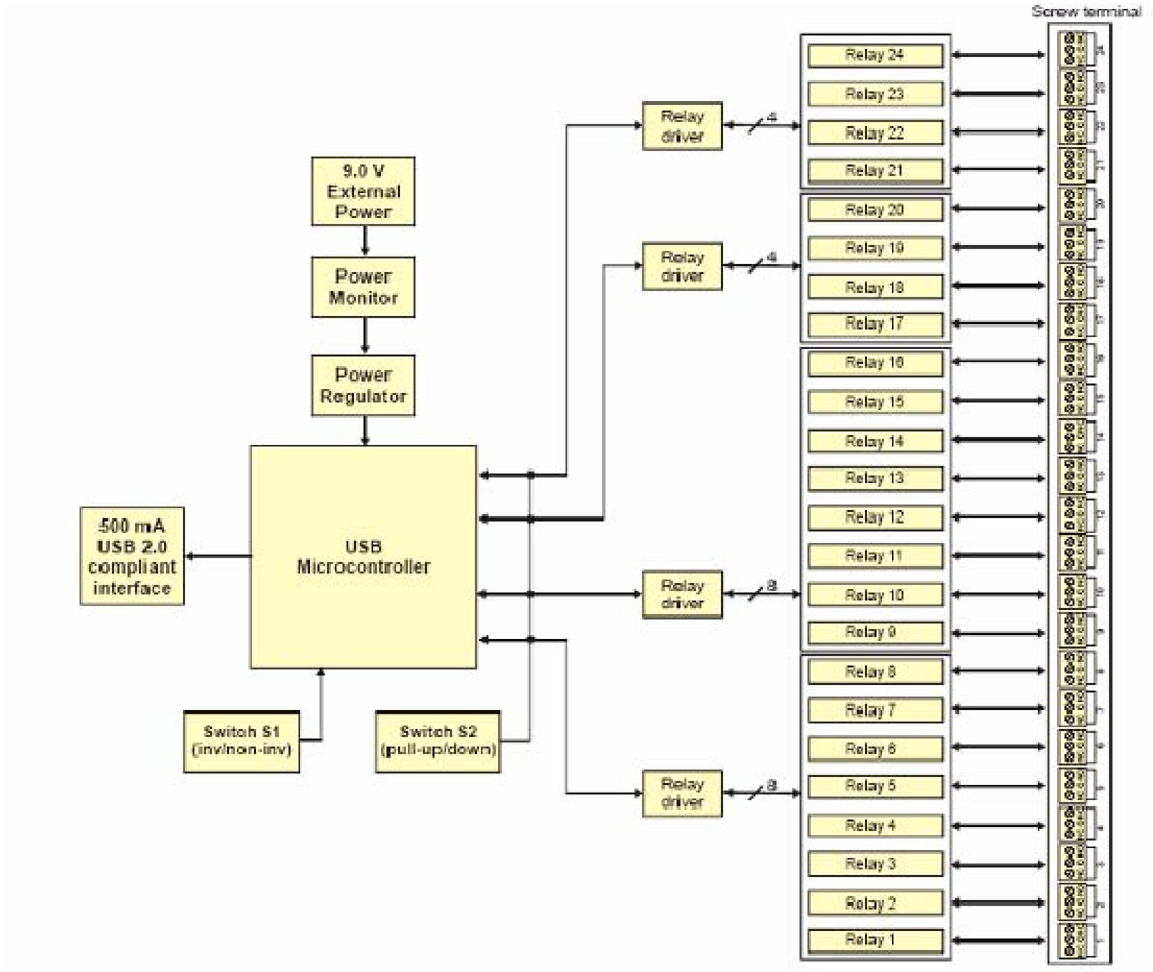
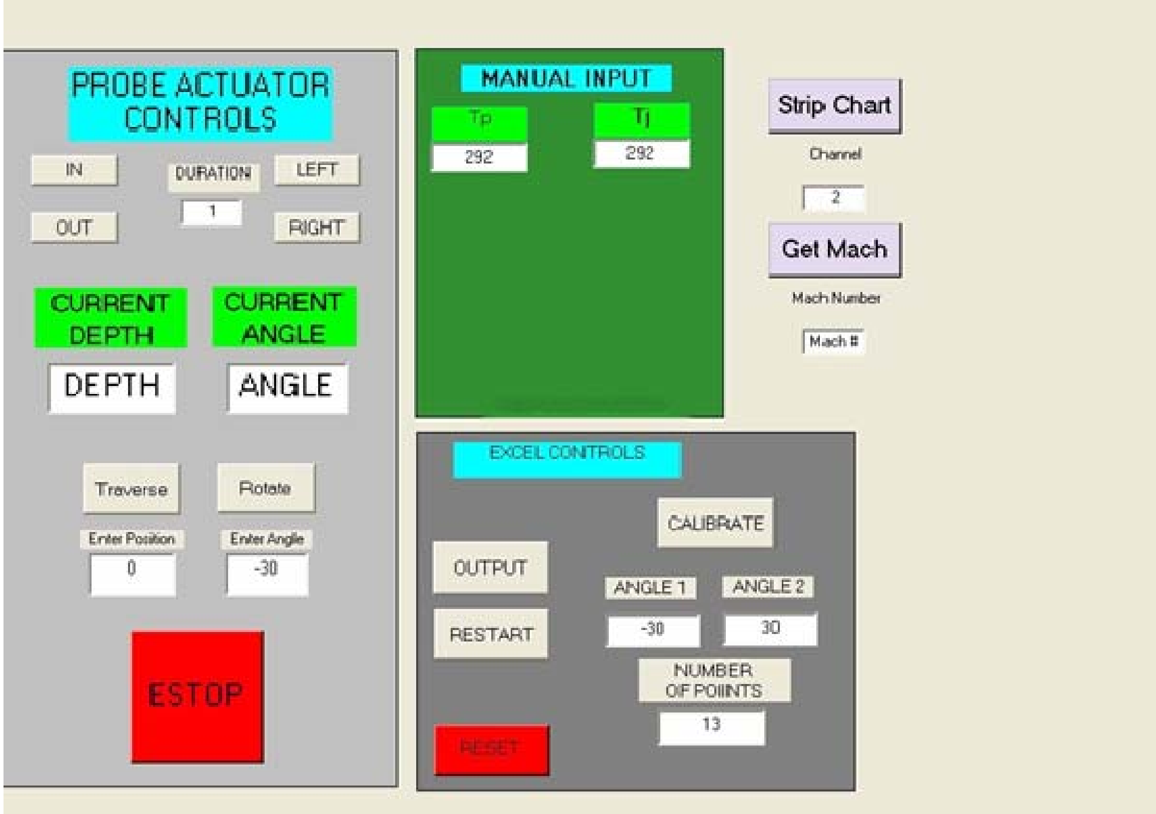


Figure 16. USB ERB24 Relay Module block diagram. (from Measurement Computing "USB-ERB24", 2005)

G. MATLAB DRIVER

Matlab Release 13, Version 6.5 was used for programming and control of data acquisition devices. Matlab was able to address the relays and guide the probe actuator through feedback controlled rotations. In addition, Matlab was able to collect and process pressure data from the transducers prior to exporting it to an Excel spreadsheet. A single Graphical User Interface (GUI) was created to perform the required functions. The GUI, shown in Figure 17, was capable of probe actuator control and data collection in the calibration procedure.



DURATION 1

Enter Position 0

Enter Angle -30

Tp 292

Tj 292

Channel 2

Mach Number Mach #

ANGLE 1 -30

ANGLE 2 30

NUMBER OF POIINTS 13

Figure 17. Matlab calibration graphical user interface.

The calibration control GUI required the manual input of the stagnation temperature of the jet, read from the HPVEE system, and the starting angle, ending angle, and number of calibration data points, or angles, as input. The function "calibration_run.m" was called from the Excel control box and outputted the required angles to use for calibration and used the feedback and control function "rotate.m", which accessed the both the USB relay and USB data acquisition (DAQ) device to position the probe actuator for each angle.

At each calibration angle, the probe would stop and the "output_to_excel.m" function called the "pressure_calibration.m" function to take pressure readings and calibrate them. The pressure data was combined with the stagnation temperature of the jet, from the GUI input, to calculate the calibration constants. The raw data along with the processed data was exported to an Excel spreadsheet. A detailed block diagram showing both probe the actuator control and the data acquisition process for the probe calibration procedure for Matlab programming is shown in Figure 18.

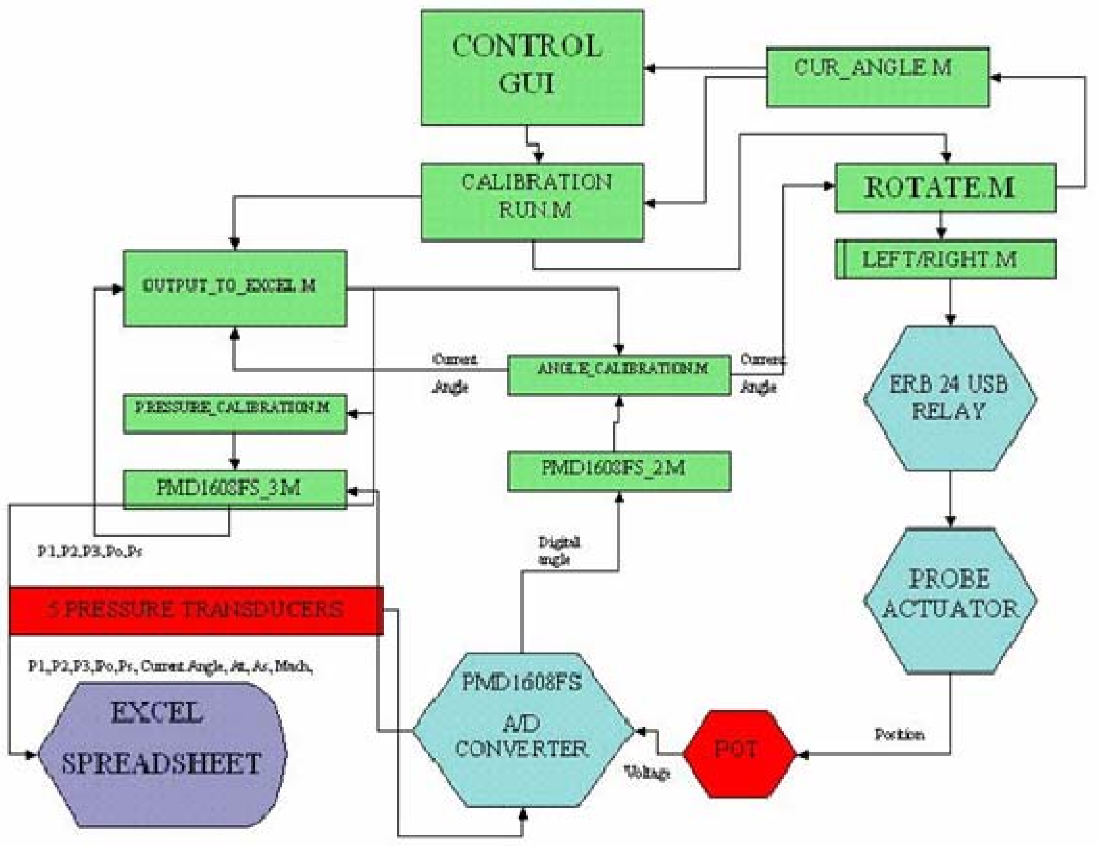


Figure 18. Matlab calibration code block diagram

The Matlab data acquisition GUI was adapted from the calibration GUI. The GUI was changed to automatically control both the angle and the traverse of the probe actuator. The Excel control box was changed to require input for the starting point and ending point for probe traversing, along with the number of points required for sampling. Once the data was entered, the "SAMPLE DATA" pushbutton sent the probe into an automatic cycle of probe traverse, rotation, and data collection. A screenshot of the Matlab data collection GUI is shown in Figure 19.

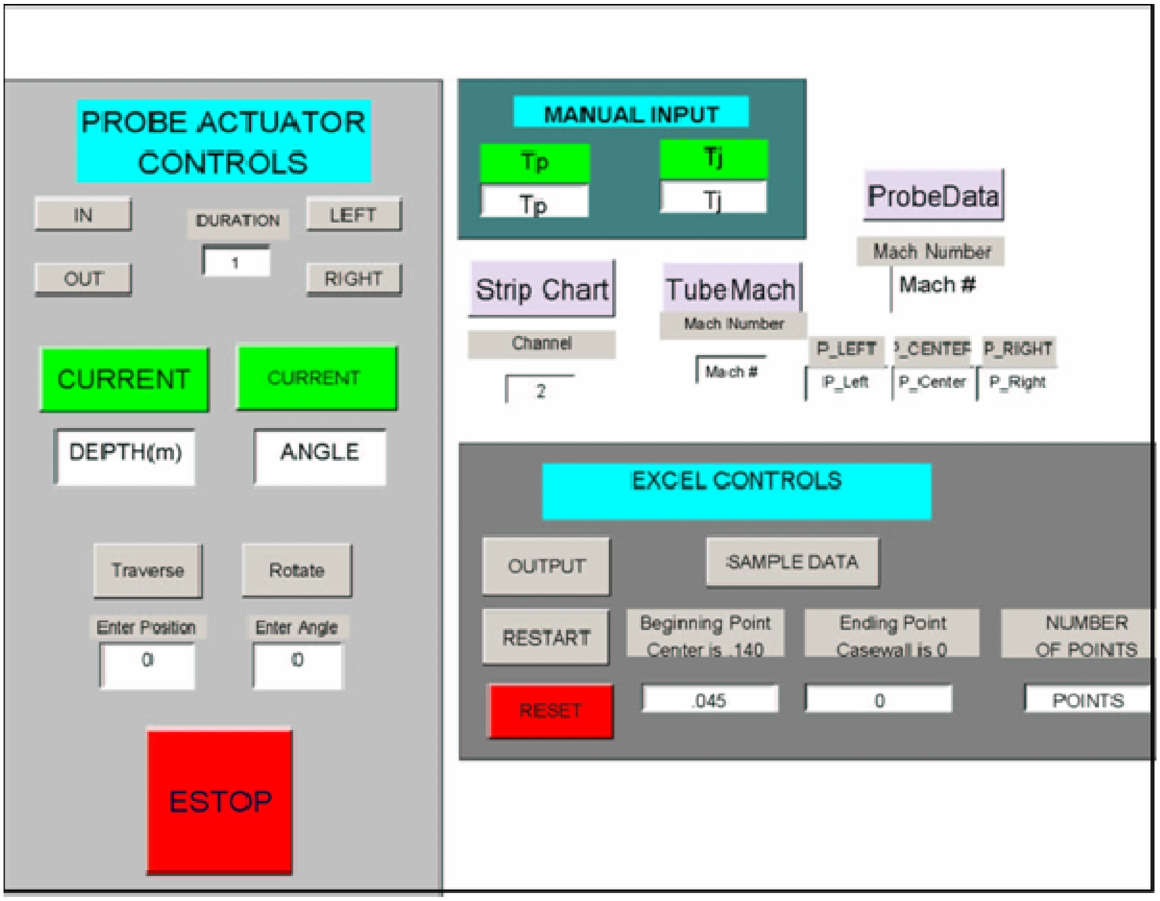


Figure 19. Matlab data collection graphical user interface.

On the data acquisition GUI, when the "SAMPLE DATA" pushbutton was depressed, the "sample_data_run.m" function processed the information that was inputted in the GUI and controlled a series of traverses to each radial position for data collection. At each radial position, the probe would sample the flow field and process the pressure data using the constants previously calculated from the pressure calibration to determine the flow relative to the pressure probe. When necessary, the "sample_data_run.m" function would input a new angle to the "rotate.m" function to perform a null yaw routine. The data would again be sampled and processed. The raw pressure and probe actuator angle data, along with processed data including the Mach number sensed and flow angle, were outputted to an Excel Spreadsheet. The Matlab experimental DAQ and feedback control block diagram is shown in Figure 20.

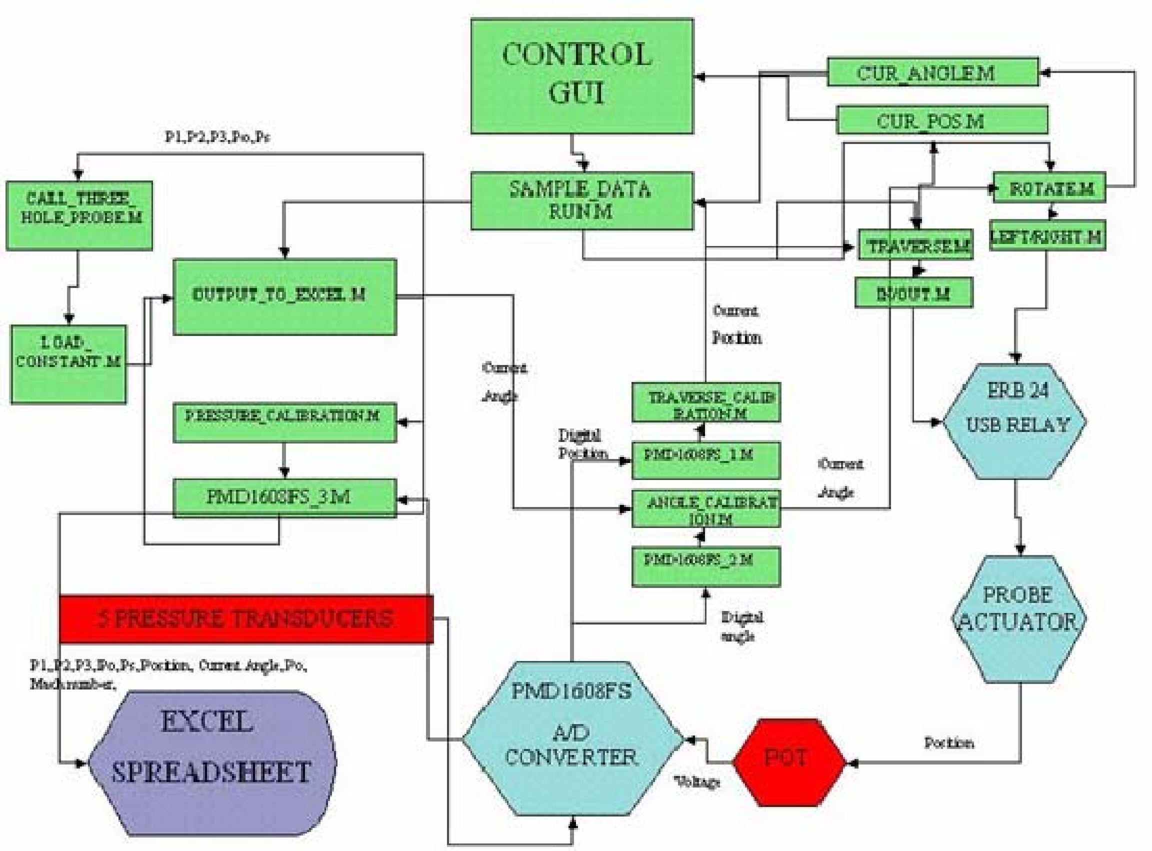


Figure 20. Matlab experimental data acquisition and feedback control code block diagram.

THIS PAGE INTENTIONALLY LEFT BLANK

IV. CALIBRATION

A. PURPOSE

The purpose of calibrating a multi-hole pressure probe was so that it could be placed in an unknown flow field to determine the magnitude and direction of the flow. In this experiment a three-hole pressure probe was calibrated by inserting it into a known flow field and rotating it through a spectrum of specific yaw and pitch angles. The data from each position angle and velocity was converted into a calibration database, utilizing a set of non-dimensional coefficients. Accurate predictions of the velocity vector in an unknown flow field could then be determined by interpolating the non-dimensional coefficients. (Johansen, Rediniotis, Jones, 2001)

B. OVERVIEW

The calibration experiment was conducted twice. In each case, the three-hole cobra probe was calibrated using the free jet test rig. A pitot-static probe, a stagnation probe, and a temperature probe were used to collect the calibration data. The probes were mounted on a bracket and situated around the free jet. The three-hole probe was mounted in the L.C. Smith actuator, capable of both axial and rotational movement.

Pressure data was initially acquired using the Scanivalve system. The Scanivalve system required an excessive amount of time, as it contained only one pressure transducer and had to step through each pressure and then wait for tube pressure to stabilize before taking a reading.

Once an initial calibration run was completed, the setup was converted to a computer logic controlled system using USB devices accessed with Matlab. Data was acquired through these devices and the run was completed. The response time of the computer feedback control system greatly reduced the required time for the three-hole probe calibration.

In addition, the Scanivalve system was replaced with five analog output pressure transducers. Each pressure transducer was connected to the USB data acquisition device. The data from the pressure transducers was verified to be accurate when compared to the previous data obtained with the Scanivalve. In both cases the resulting calibration coefficients for the pitch angles were not sufficiently discriminatory and pitch measurements were found to be unreliable, hence pitch was not used in the experimental data acquisition for the three-hole probe.

C. CALIBRATION PROCEDURE

The probe actuator containing the three-hole probe was attached to the free jet test assembly previously shown in Figure 12. The jet nozzle had a diameter of 10.8cm (4.25inches) and the pitot-static and stagnation probes were both placed at the same radial distance from the center of the flow 2.86cm (1.125inches).

The three-hole probe was placed into a Mach 0.5 flow and rotated until a manometer showed equal pressures on ports 2 and 3 shown in Figure 21. The center hole was designated "1" in accordance with normal industry standards. When facing into the flow, the left side was designated "2" and the right side was designated "3."

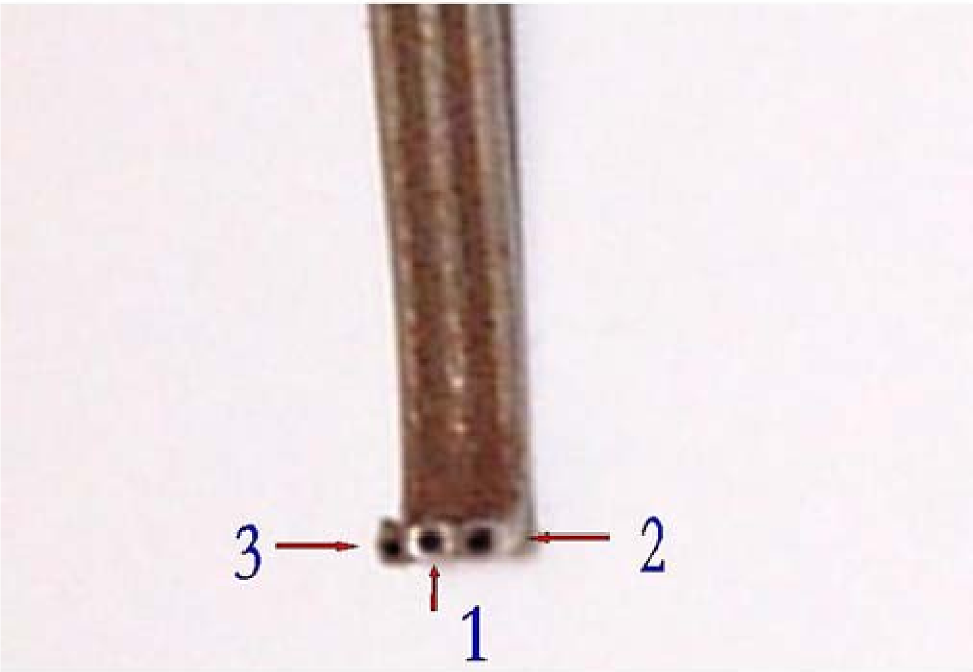


Figure 21. Three-hole probe numbered.

Once the manometer showed equal pressures, the pressure tubes were removed from the manometer and connected to the Scanivalve ports 5 and 7. The Scanivalve system verified ports 2 and 3 read equal pressures on the HPVEE system. The angle at which the outer pressures were equal was designated the "zero" angle and used as a reference point for calibration.

A post-processing program was created using the HPVEE system to determine the calibration constants for the three-hole probe. Calibration data was obtained using the HPVEE/Scanivalve system described by Grossman. (Grossman, 1997) Screen captures of the HPVEE system are shown in Appendix D.

In the free jet test assembly, the pitot-static and stagnation probes were used to calculate the Mach number. The Mach number was increased and decreased by adjusting the mass flow rate through the nozzle. This was conducted by throttling the dump valve on the Allison Chalmers compressor for Mach numbers above 0.3 and by throttling the control valve in the cell for below Mach 0.3.

The calibration was conducted by rotating the probe from –30° to 30° with a manual electric control switch. Pressure and temperature readings were taken at five-degree increments and exported to an Excel database. Each complete rotation was conducted for Mach numbers 0.85, 0.7, 0.5, 0.3, .1 and 0.05 with the probe perpendicular to the flow.

D. CALLIBRATION PROCEDURE WITH AUTOMATED CONTROLS AND INDIVIDUAL PRESSURE TRANSDUCERS

The probe actuator was adapted to a feedback control system using the USB controlled relays, and a USB A/D converter. The feedback was provided through the Measurement Computing (MC) Analog Input device PMD-1608FS, which measured the voltage output of the potentiometers on the probe actuator and was accessed by Matlab via a USB cable. Control of the actuator was also provided via USB cable, with Matlab accessing relays on the MC ERB-24 to rotate and traverse the probe. Matlab control programming can be found in Appendix G. The feedback control system and data-acquisition wiring diagram is shown in Figure 22.

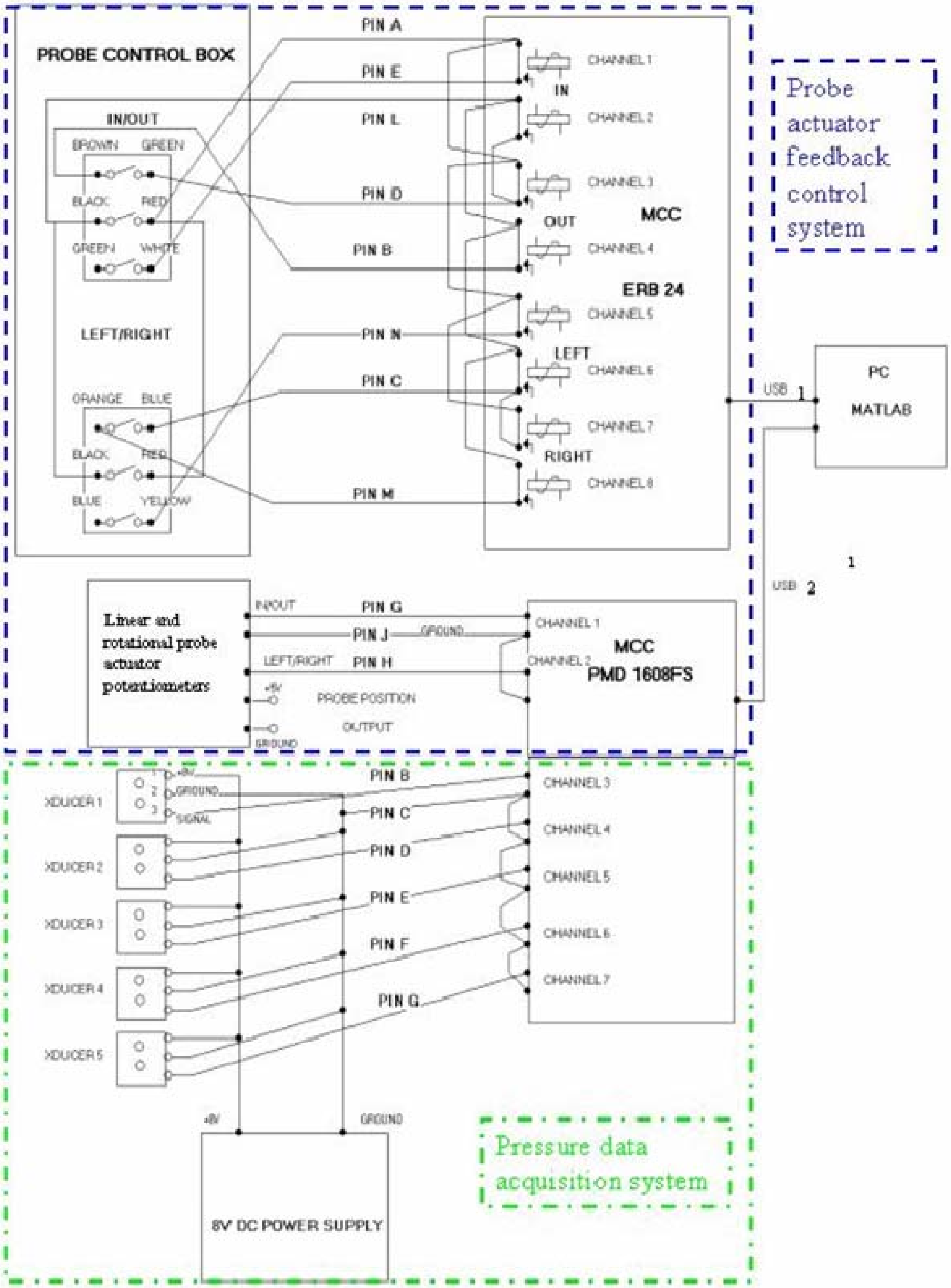


Figure 22. Probe actuator and data acquisition wiring diagram.

Omega Engineering PX-138 Pressure transducers were integrated into the pressure lines to replace the Scanivalve system. Five pressure transducers were used to provide individual pressure outputs for the three-hole, static, and stagnation pressure probes. Each pressure transducer produced an analog signal (voltage), which was digitized by the PMD-1608FS and converted to engineering units (Pascals) using a Matlab calibration program. The Matlab program also calculated the probe calibration constants and exported all of the data to an Excel Spreadsheet.

The transducers were calibrated by creating a table of voltage outputs for known pressures. The transducers had two ports to measure a differential pressure. One port of each transducer was connected to a pressure standard. To verify accuracy, the calibration pressure was then connected to the other pressure port on each transducer and the outputs were once again tabulated. Once complete, the pressure transducers were disconnected from the calibration pressure and connected to the pressure probes. The response of the transducer was found to be nearly linear for both sides, positive and negative pressures. A linear interpolation method in Matlab was used to find the calibration for pressures. When the PX-138 was used for calibration, it was commanded to take 30 samples at 500hz, average the results, and repeat this process four times. The four results were again averaged to create the output of each pressure transducer. Calibration data and Matlab programming for the PX-138 transducers can be found in Appendix F.

The calibration experiment was repeated using the control system and PX-138 pressure transducers. Using a feedback loop for angle position and automatically rotating through the -30° to 30° the time required for the experiment was greatly reduced. Data obtained for calibration was comparable to that obtained by the Scanivalve system.

E. CALIBRATION EQUATIONS

1. Purpose

The purpose of the calibration calculation was to determine the non-dimensional variables for each angle. The total pressure coefficient (A_{t_probe}), and the static pressure coefficient (A_{s_probe}) had to be determined for each angle in a flow field of a known velocity to be later used to determine the flow angle in an unknown field. A sample calculation of the calibration constants is shown in Appendix E to demonstrate the calibration process. The calibration calculations were adapted from five-hole probe calibration calculations. (Johansen, et al., 2001)

2. Setup and Nomenclature

The center hole was designated "1" in accordance with standard industry designation. When facing into the flow, the left side was designated "2" and the right side was designated "3." Pitch and yaw angle definitions are shown in Figure 23.

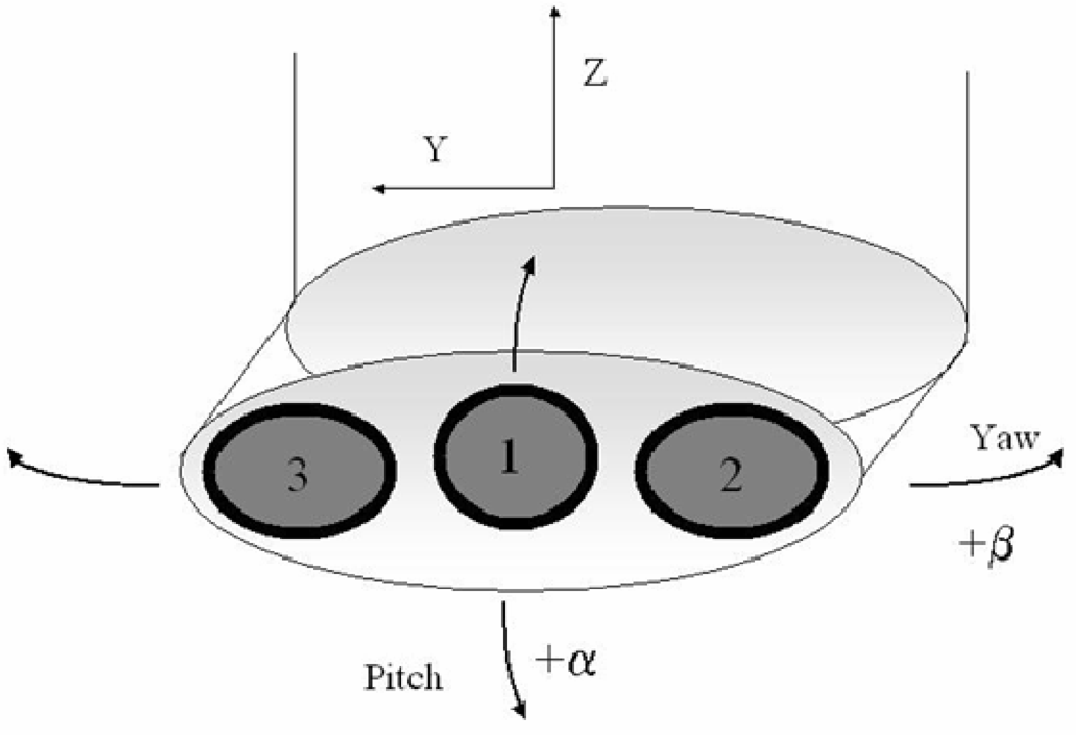


Figure 23. Three-hole probe pitch and yaw angle definitions.

Table 2. Calibration nomenclature.

Symbol	Definition
β	Yaw Angle
P_1	Center Probe Pressure
P_2	Port or Left Side Probe Pressure when facing flow
P_3	Starboard or Right Side Probe Pressure when facing flow
P_{oj}	Jet Stagnation Pressure
P_{sj}	Jet Static Pressure
T_p	Jet Stagnation Temperature
γ	Specific Heat Ratio
C_p	Constant Pressure Specific Heat
q_probe	Probe Pseudo Dynamic Pressure
V_jet	Jet Velocity
C_{p_1}	Left High Angle Flow Discriminator
C_{p_2}	Center Low Angle Flow Discriminator
C_{p_3}	Right High Angle Flow Discriminator
A_{t_Probe}	Probe Total Pressure Coefficient
A_{s_Probe}	Probe Static Pressure Coefficient
MachNo	Mach Number Calculated with Jet Static and Stagnation Pressure Probes

3. Calibration Equations

The pseudo dynamic pressure, q_probe, provided a single pressure value for each data point for the probe and was given as:

$$\text{q\_probe} := P_1 - \frac{\left(P_2 + P_3\right)}{2} \qquad 4.1$$

The jet velocity was calculated with the static and stagnation probes and is given as:

$$\text{V\_jet} := \sqrt{\left[1 - \left(\frac{P_{sj}}{P_{oj}}\right)^{\frac{(\gamma-1)}{\gamma}}\right]} \cdot \sqrt{2 \cdot C_p \cdot T_p} \qquad 4.2$$

The angle flow discriminators, C_{p_1}, C_{p_2}, and C_{p_3}, provided a sense of the magnitude of the direction of the flow, relative to the probe.

$$C_{p\_1} := \frac{\left(P_1 - P_2\right)}{\text{q\_probe}} \qquad 4.3$$

$$C_{p\_2} := \frac{\left(P_1 - P_3\right)}{\text{q\_probe}} \qquad 4.4$$

$$C_{p\_3} := \frac{\left(P_2 - P_3\right)}{\text{q\_probe}} \qquad 4.5$$

The total and static pressure coefficients provided the measurable data to be used during flow field surveys.

$$A_{t\_Probe} := \frac{\left(P_1 - P_{oj}\right)}{q\_probe} \qquad 4.6$$

$$A_{s\_Probe} := \frac{q\_probe}{\left(P_{oj} - P_{sj}\right)} \qquad 4.7$$

The Mach number was also calculated from the static and stagnation probe readings:

$$MachNo := \sqrt{\left[\left(\frac{P_{oj}}{P_{sj}}\right)^{\frac{(\gamma-1)}{\gamma}} - 1\right] \cdot \left(\frac{2}{\gamma - 1}\right)} \qquad 4.8$$

The calibration constants for each speed and angle were tabulated in an Excel spreadsheet. When the table was complete, the data was stored and placed in a binary Matlab format (MAT) file for use in subsequent flow field surveys.

E. PREDICTION OF VELOCITY AND YAW ANGLE

To determine the unknown velocities and yaw angles, the Matlab function GRIDDATA3 was used. The calibration data was accessed through a saved Matlab workspace text MAT file and loaded each time the flow field survey was conducted. GRIDDATA3 interpolated hyper-surfaces using tessellation-based linear interpolation based on a Delaunay triangulation of the data. (The Mathworks Inc, 2002) Interpolation constants and Matlab programming are contained in Appendix I.

F. COMPARISON OF SCANIVALVE AND PX-138 DATA

The Scanivalve and PX-138 systems created similar calibration databases. However, the PX-138 system was much quicker, and produced smoother calibration curves for the As and At calibration constants, especially in the lower Mach numbers. The Scanivalve system produced data with more variations when the flow was below Mach 0.5. Three-dimensional graphs of the Scanivalve and PX-138 calibration surfaces are shown in Figures 24-27.

The calibration curves for the transducer data are visually smoother than that of the Scanivalve system. However, in comparing the calibration surface graphs, consideration must be given to the fact that the Scanivalve system was calibrated to a Mach number of 0.05 and the PX-138 Transducers were only calibrated to a Mach number of 0.066. The data was not calculated as low for the PX-138 transducers because the Scanivalve data was inconsistent below Mach 0.1 and it was not believed the transducers could be calibrated for very low flow fields as well. Further calibration details, including calibration calculations and variables, software programs used, calibration data, and calibration curves, are shown in Appendices E-I.

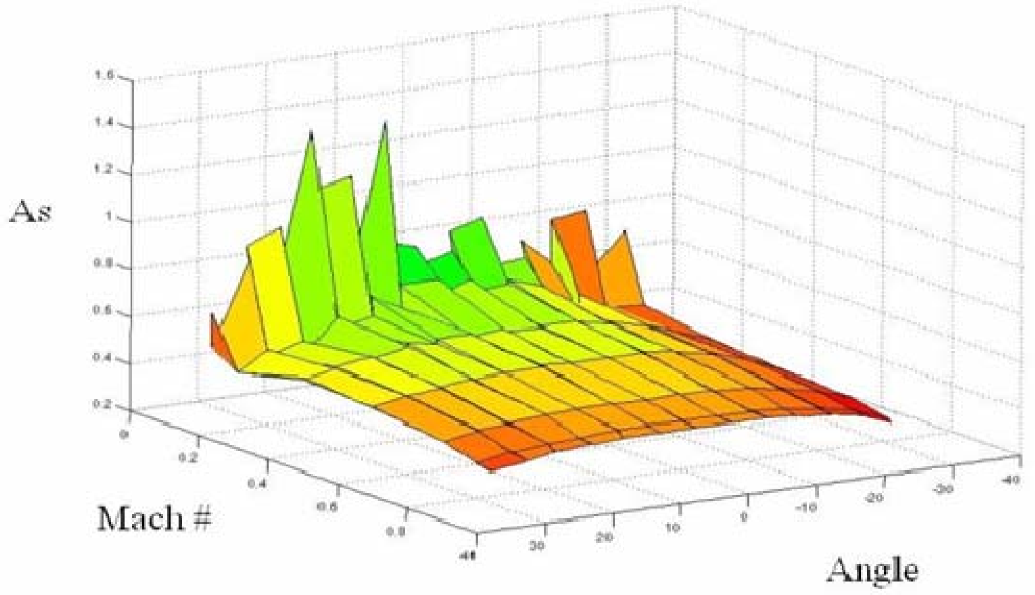


Figure 24. Scanivalve "As" Calibration Surface.

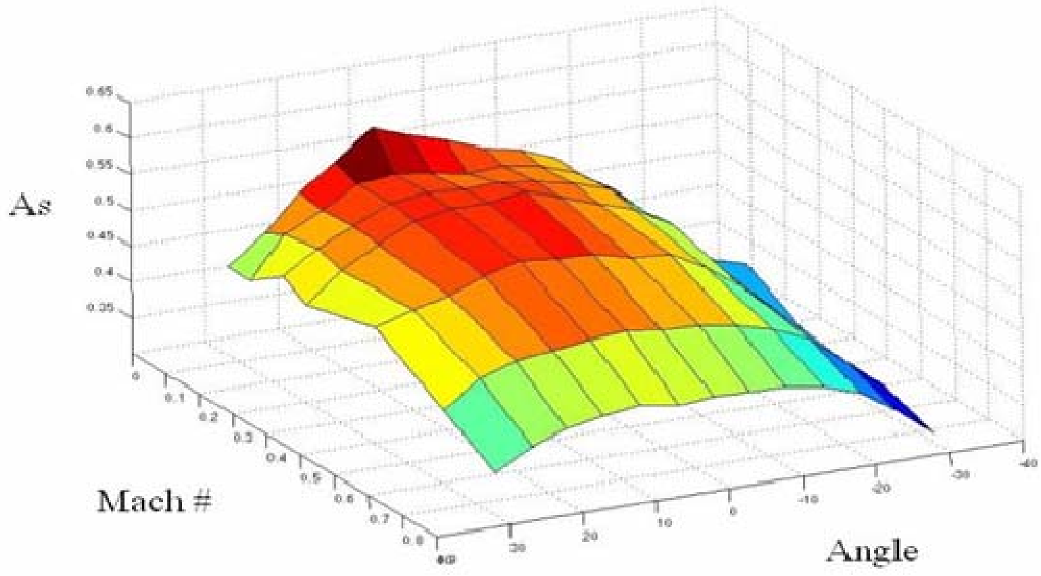


Figure 25. PX-138 Transducer "As" Calibration Surface.

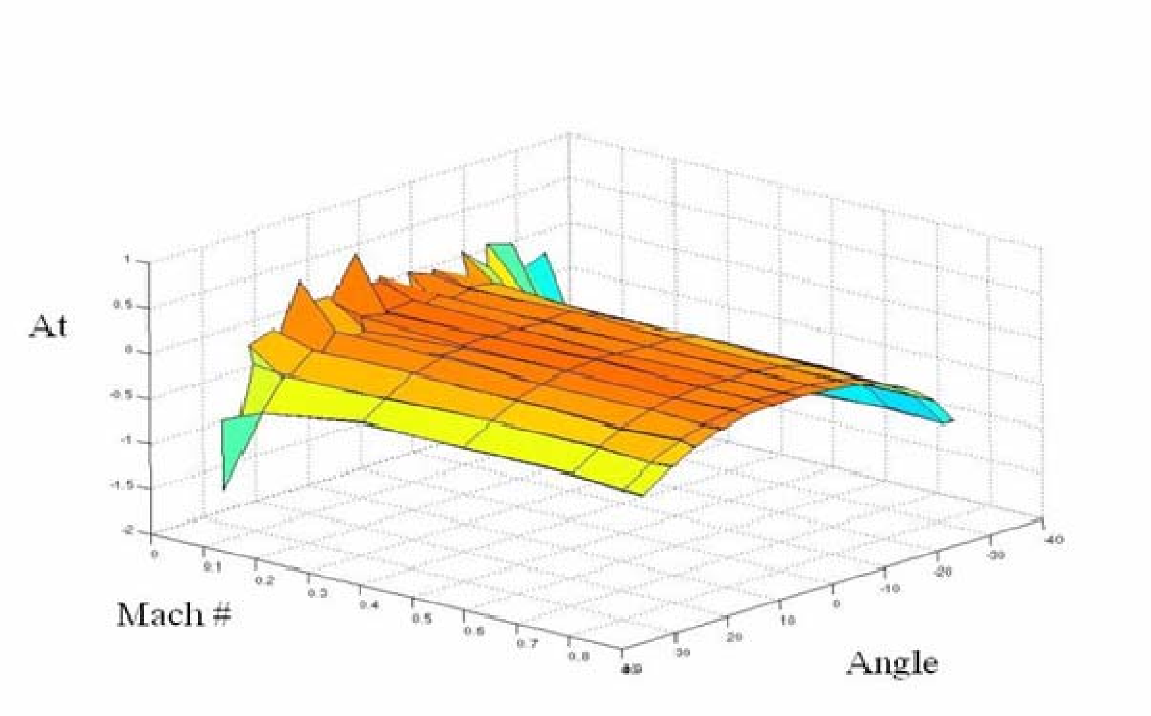


Figure 26. Scanivalve "At" Calibration Surface.

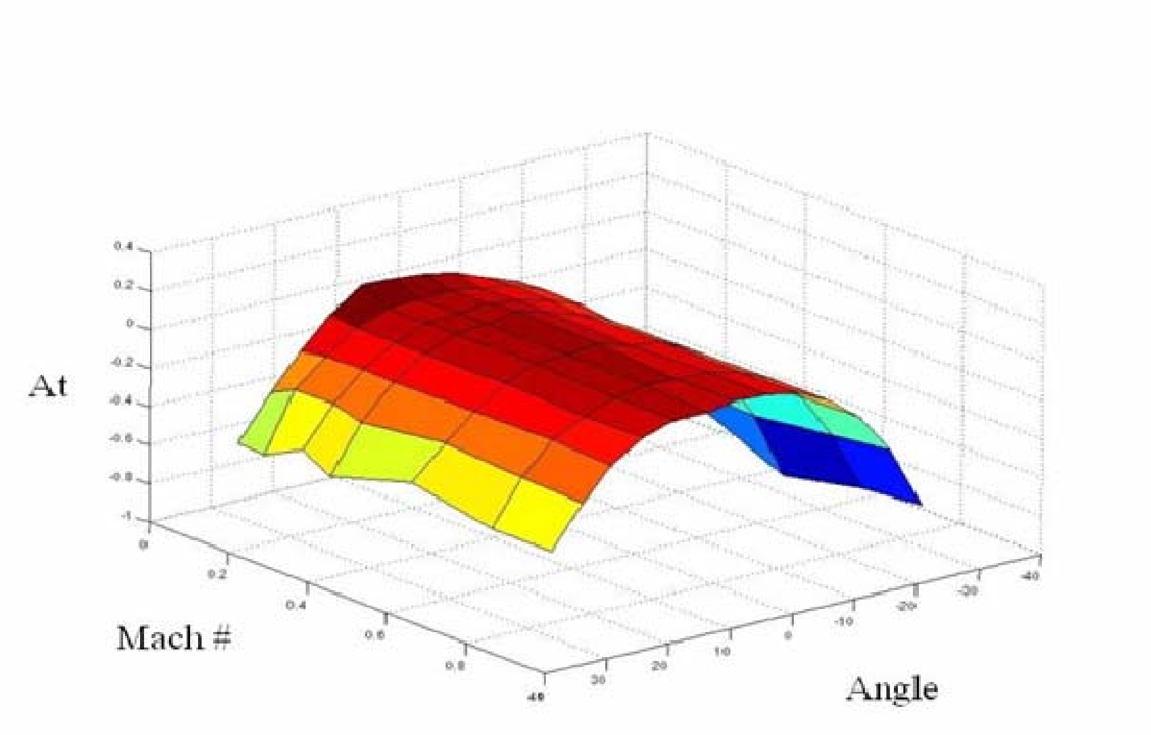


Figure 27. PX-138 Transducer "At" Calibration Surface.

V. TEST RUN PROCEDURE

A. OVERVIEW

The L.C. Smith probe actuator containing the three-hole probe was attached to the steel case wall of the Transonic Compressor. Test runs were conducted on both upstream and downstream positions. The probe was inserted into the unknown flow at various depths and data was collected at each position. The process was repeated for different speeds and mass flow rates.

The data collected is the baseline data to determine the normal operational parameters of the rotor at various speeds and mass flow rates. From this data, experiments following can calculate the effects of steam ingestion on stall margins and rotor performance.

B. TYPICAL TEST PROCEDURE

The upstream position utilized hole one while the downstream position used hole four. The L.C. Smith actuator was firmly attached to the mounting bracket on the steel case wall and the probe was controlled through Matlab programming with a remote PC as described in Chapter III.

For the upstream position, 15 data points were taken. The Matlab program used a cosine(π/2) function to identify the points to sample data. The cosine(π/2) function was projected on to the distance between center and case wall so that the frequency of data points increased as the probe approached the case wall. The probe would begin at the center of the flow cylinder and incrementally traverse towards the case wall. The increased number of data points closer to the case wall improved the resolution of the boundary layer. Positions of the cosine(π/2) function are shown in Table 3.

For the downstream position, 15 data points were also taken. For this position, a cosine(π) function was used to sample data more frequently closer to the hub and case wall. A depth micrometer was used to identify the exact distance from the hub to the case wall. The probe would begin at an approximate 1.5mm offset from the hub, and traverse towards the case wall. The cosine(π) function was projected on to the distance between hub and case wall. Positions of the cosine(π) function are also shown in Table 3 below.

Table 3. Probe sampling positions

Position	Upstream Cos(π/2) in cm	Downstream Cos(π) in cm
1	13.97	4.5
2	12.41	4.44
3	10.86	4.28
4	9.36	4.01
5	7.91	3.65
6	6.54	3.23
7	5.26	2.75
8	4.09	2.25
9	3.05	1.75
10	2.14	1.27
11	1.38	.85
12	.78	.49
13	.35	.22
14	.09	.06
15	0	0

Null yawing was used in the downstream position. The angle of flow varied from the hub to the case wall and with speed and mass flow rate. To ensure the accuracy of the measurement, an additional loop in the Matlab programming was created. Logic was added so that if the measured angle of the flow were greater than 5° from the angle of the probe, the probe actuator would turn the probe into the flow to null yaw the probe. The calibration of the probe was assumed to be very accurate within the near linear area between ±15° and satisfactory results were obtained. However, the final data point closest to the case wall occasionally read a large angle close to + or - 90°. This caused the Matlab program to fail to output the final position as the angle was out of normal parameters. It was believed this erroneous data is caused by the probe tip's proximity to the probe's insertion hole in the case wall and possible swirling of flow in that region.

Each test run was conducted at a constant speed, with mass flow rate adjusted for each set of data points. Runs were conducted at 70, 80, 90, and 100 percent speed. 100 percent speed corresponded to the design speed of 27,085 revolutions per minute.

On each test run, the first set of data was conducted with the compressor throttle fully open. This setting was labeled as the "Choke" setting. For the second run, the mass flow rate was reduced to the predetermined "Peak Efficiency" setting for the each speed in accordance with the Mass flow-Isentropic efficiency graph contained in Appendix B. The final run of each series was with a reduced flow corresponding to the "near stall" mass flow rate. Table four identifies the rotor speeds and mass flow rates used for calculations.

Table 4. Rotor speed and mass flow rates.

Speed		Mass Flow Rates (Kg/sec)		
Speed	Speed (RPM)	Choke	Peak Efficiency	Near Stall
70%	18959.5	5.95	5.58	4.61
80%	21668	6.73	6.47	5.49
90%	24376.5	7.39	7.24	6.21
100%	27085	8.09	7.97	7.69

THIS PAGE INTENTIONALLY LEFT BLANK

VI. RESULTS AND DISCUSSION

A. ANALYSIS

1. Upstream Data

Inlet surveys were conducted at 70, 80, 90, and 100 percent speed. The radial Mach number distributions are shown in Figures 28-31 with yaw angle and stagnation pressure in Appendices J and K.

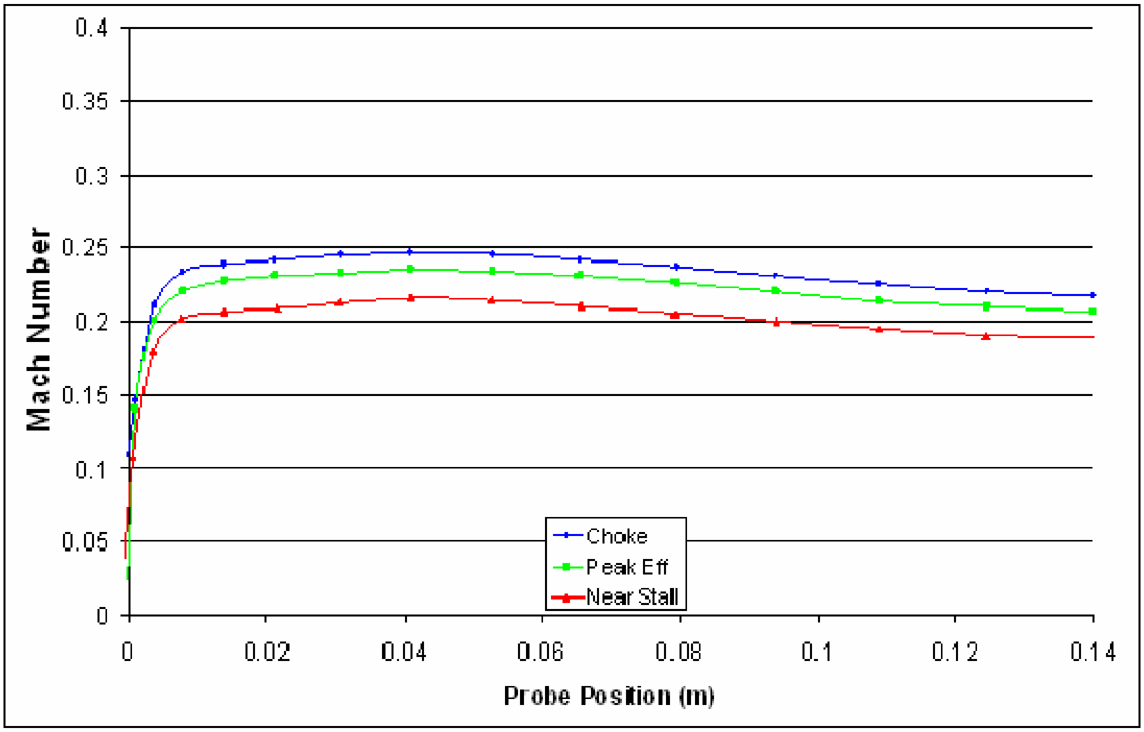


Figure 28. Inlet Mach number at 70% speed.

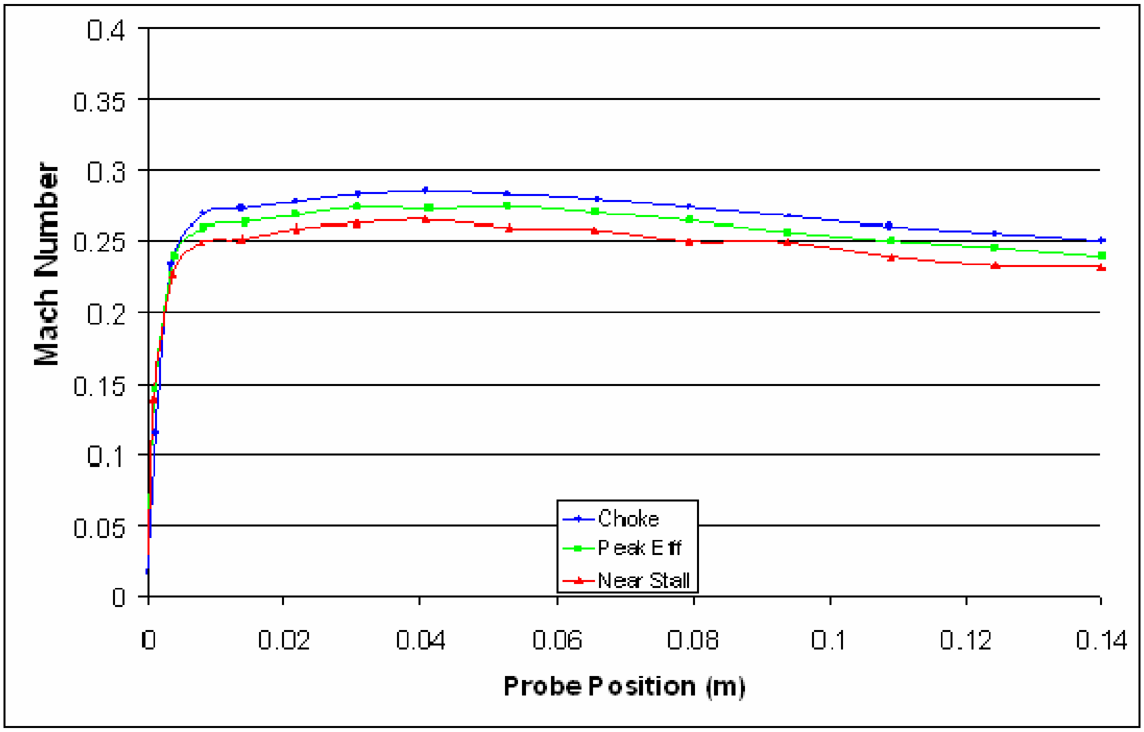


Figure 29. Inlet Mach number at 80% speed.

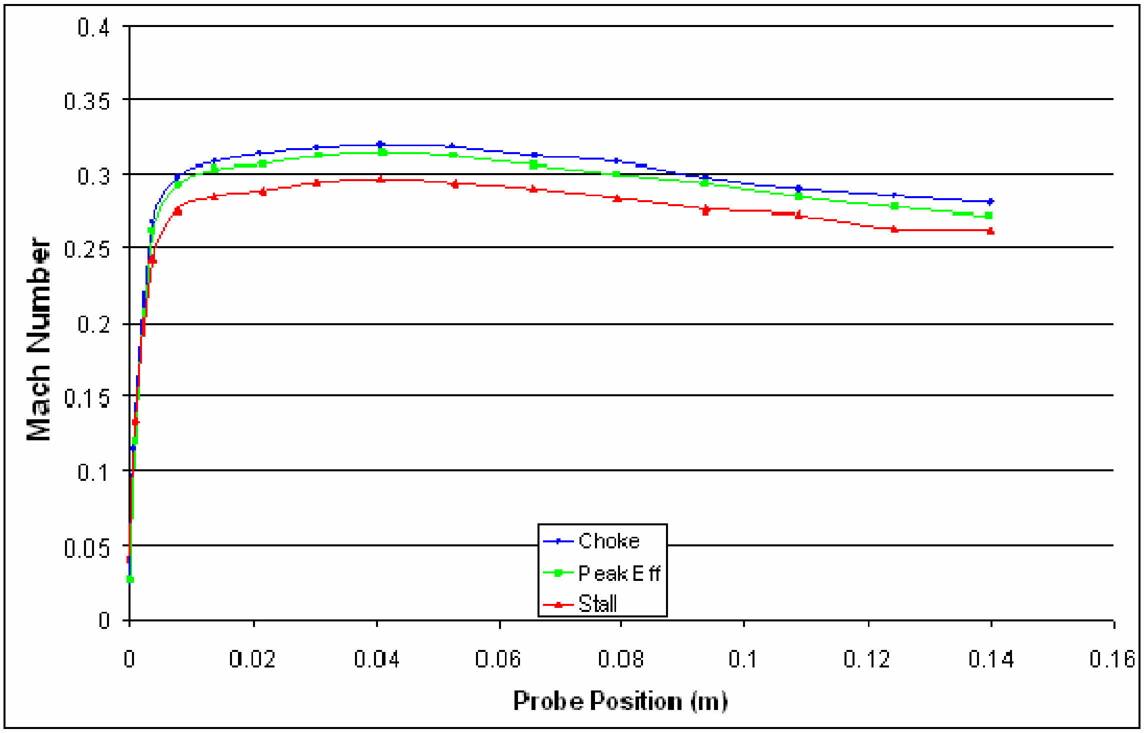


Figure 30. Inlet Mach number at 90% speed

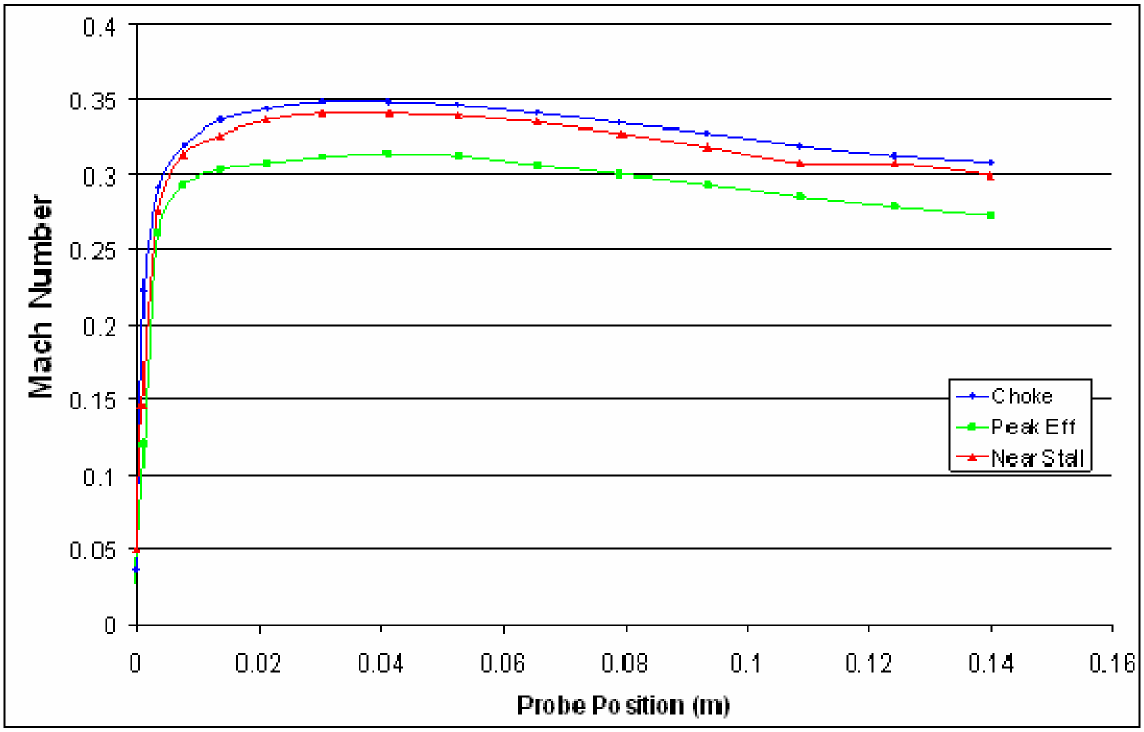


Figure 31. Inlet Mach number at 100% speed.

2. Downstream Data

Exit surveys were conducted downstream of the rotor. Results are shown in Figures 32-39 for rotor speeds of 70, 80, 90, and 100% with Mach number and flow angle plotted against the hub-to-tip radius ratio. Graphs of the downstream stagnation pressure ratio for each speed are shown in Appendix L.

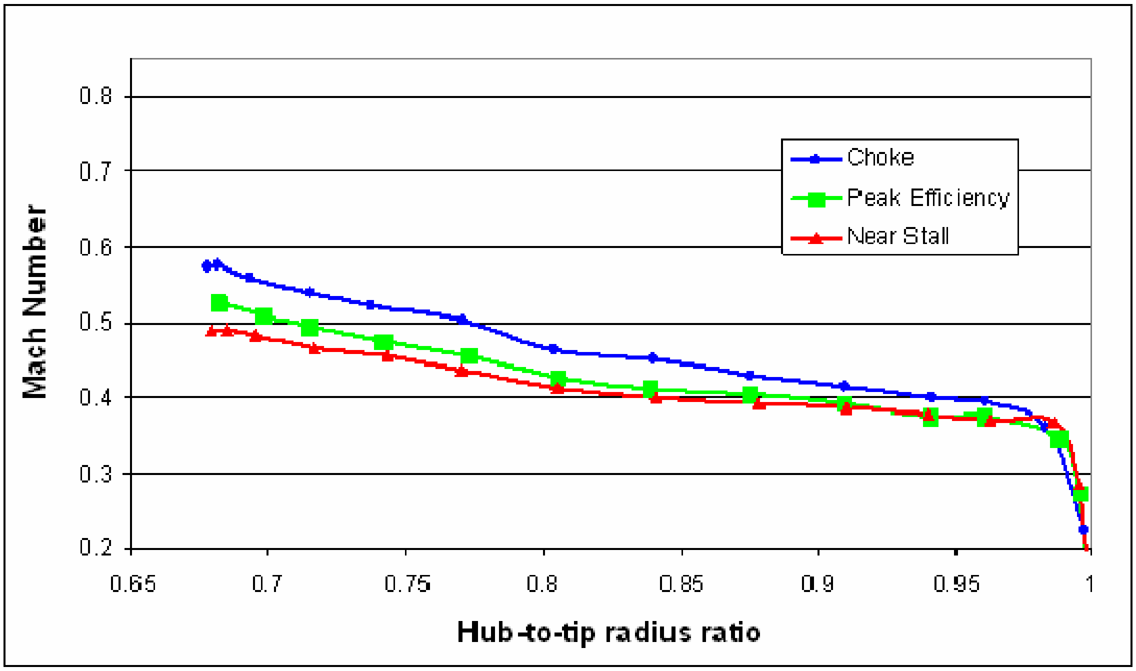


Figure 32. Exit Mach number, 70% speed.

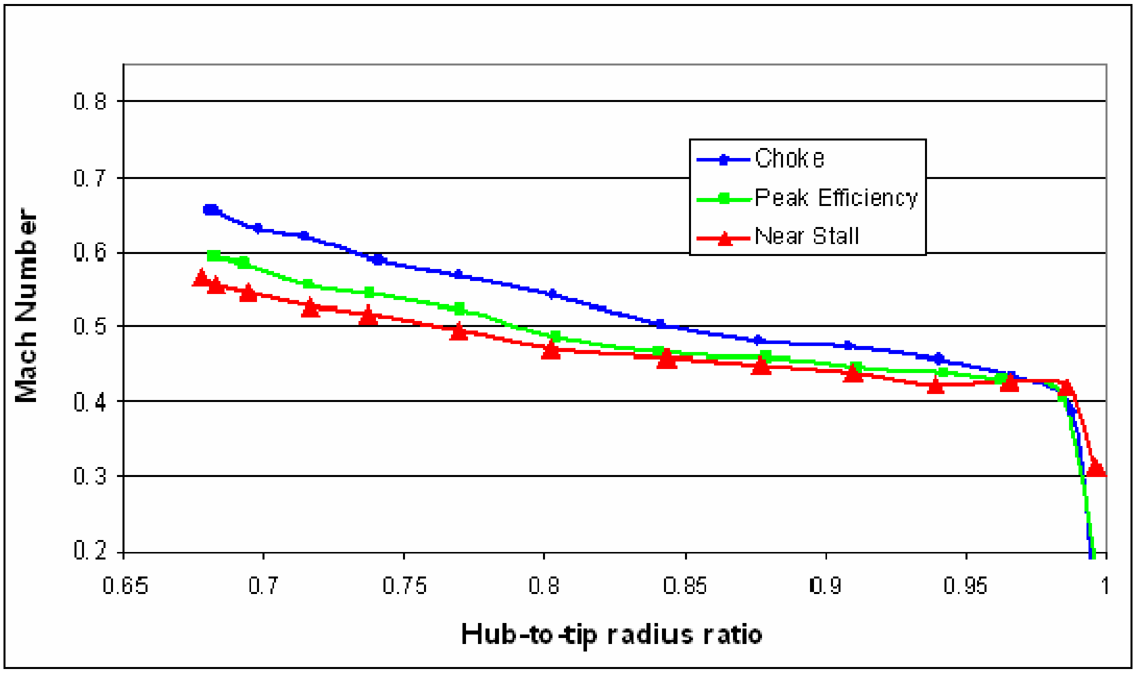


Figure 33. Exit Mach number, 80% speed.

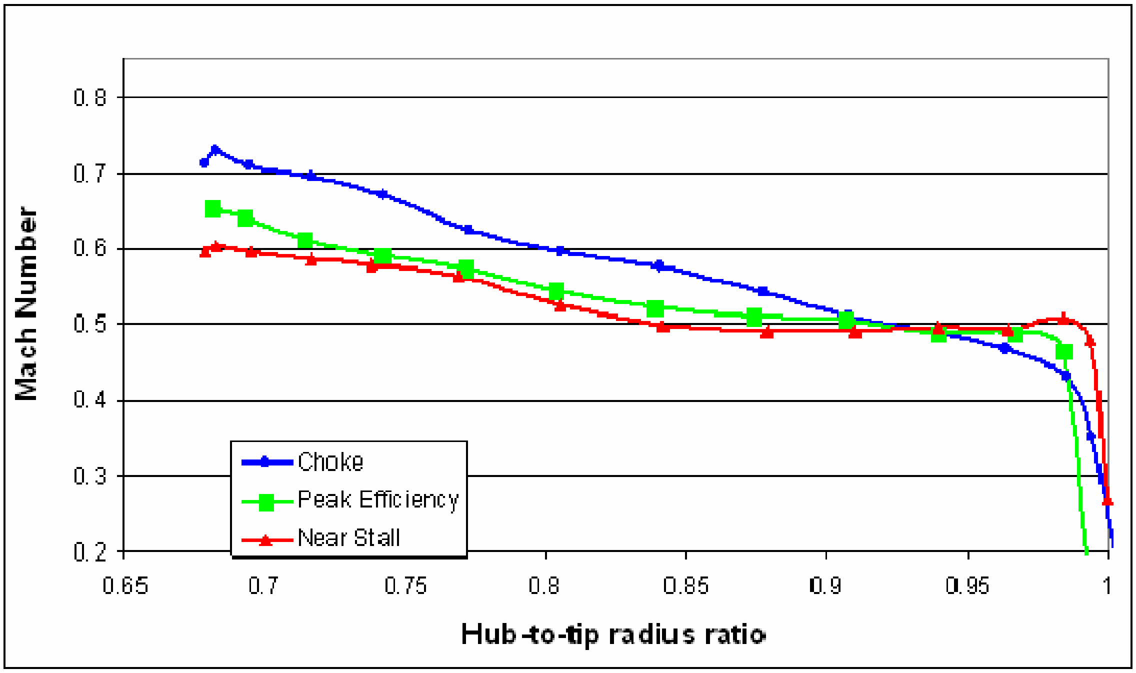


Figure 34. Exit Mach number, 90% speed.

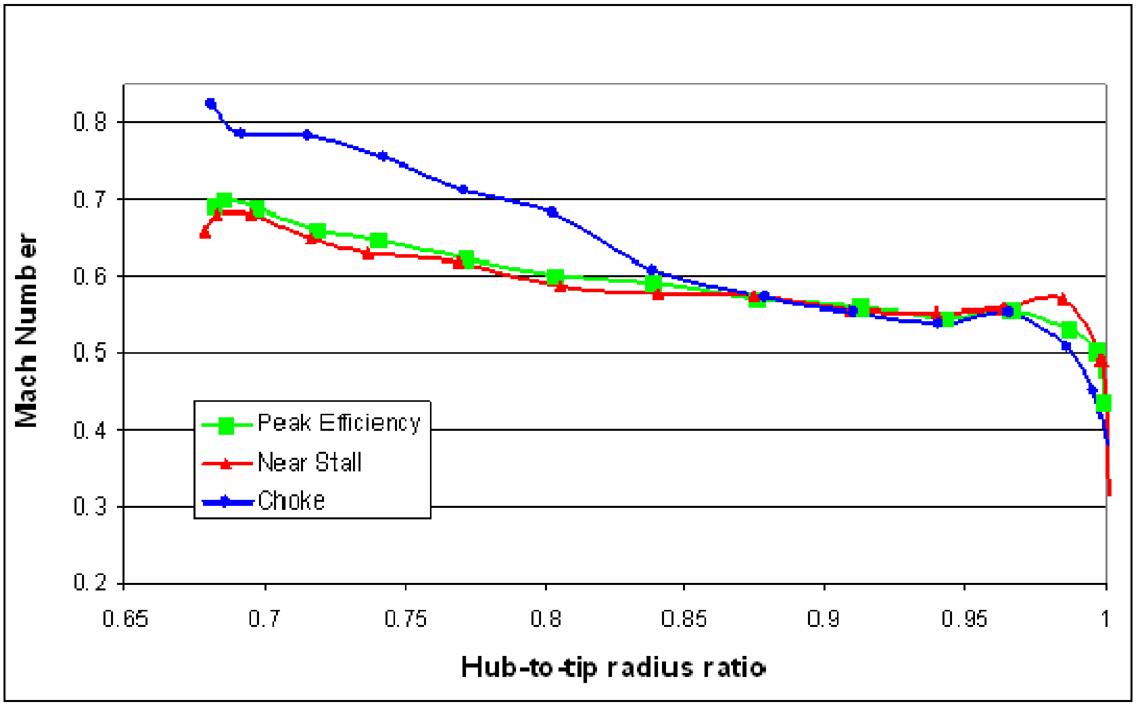


Figure 35. Exit Mach number, 100% speed.

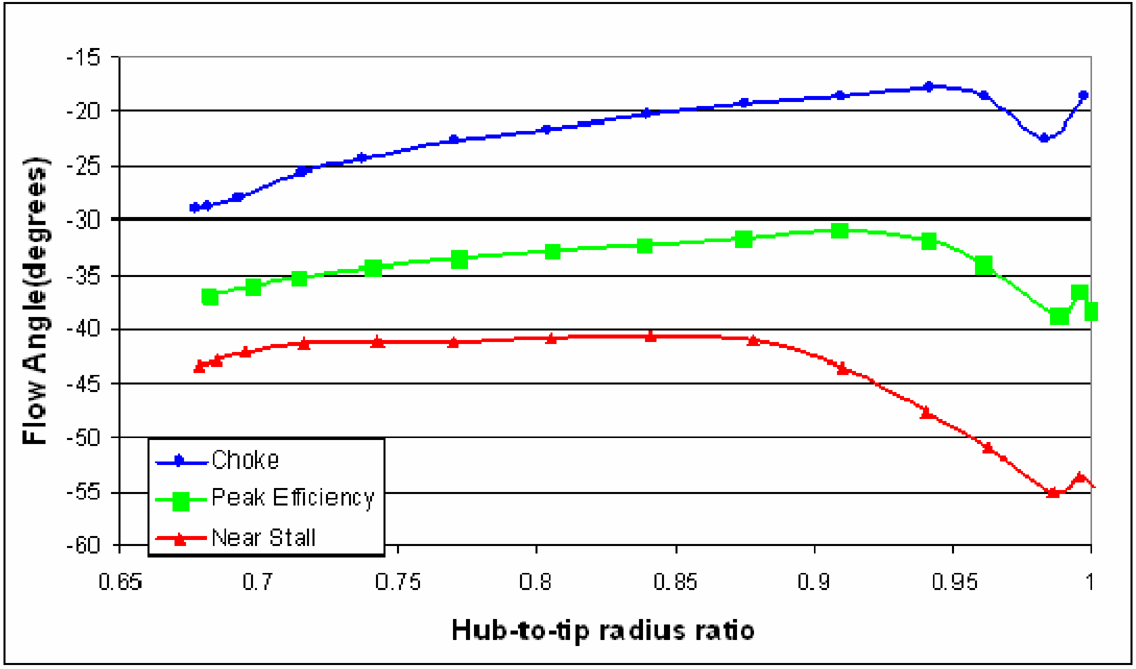


Figure 36. Exit flow angle, 70% speed.

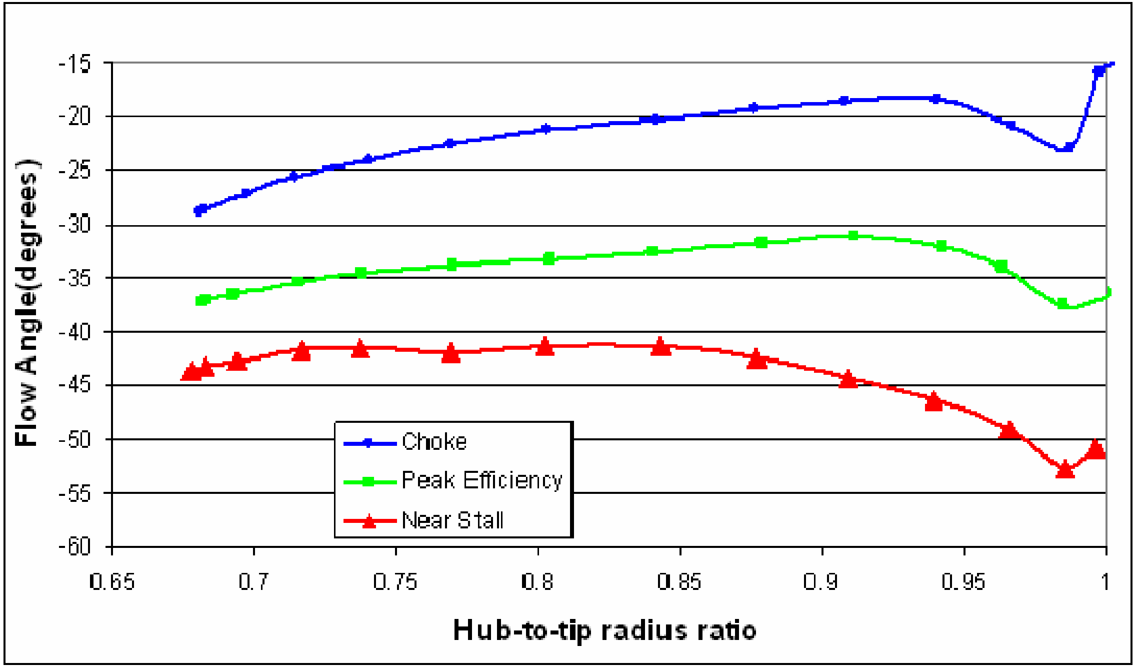


Figure 37. Exit flow angle, 80% speed.

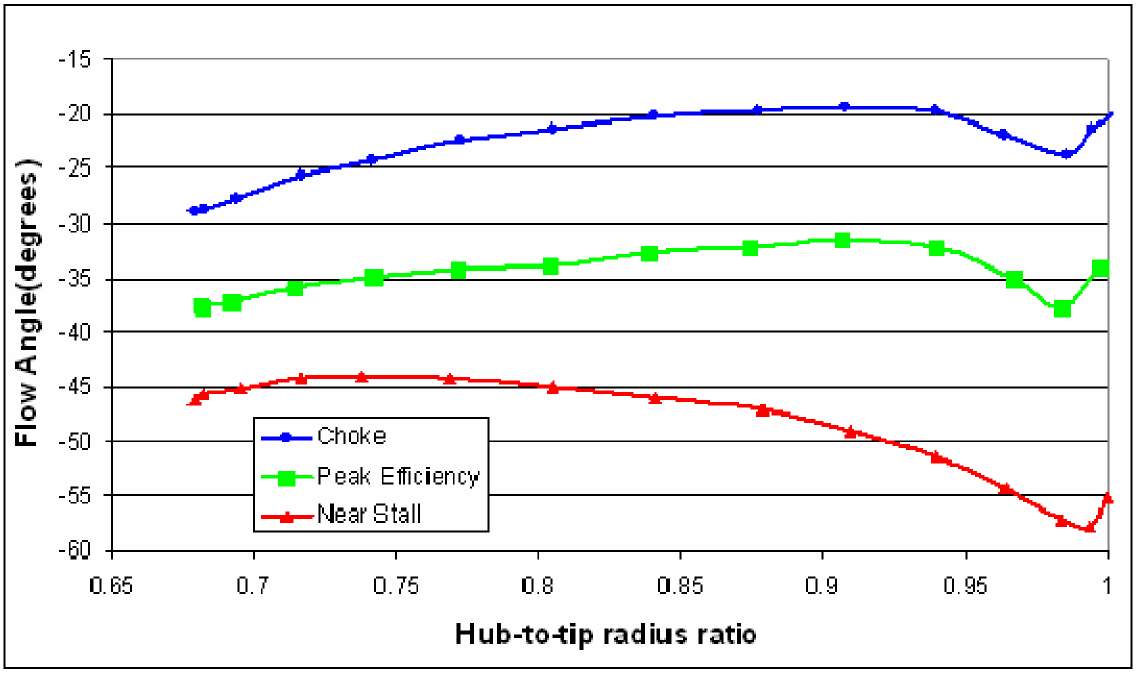


Figure 38. Exit flow angle, 90% speed.

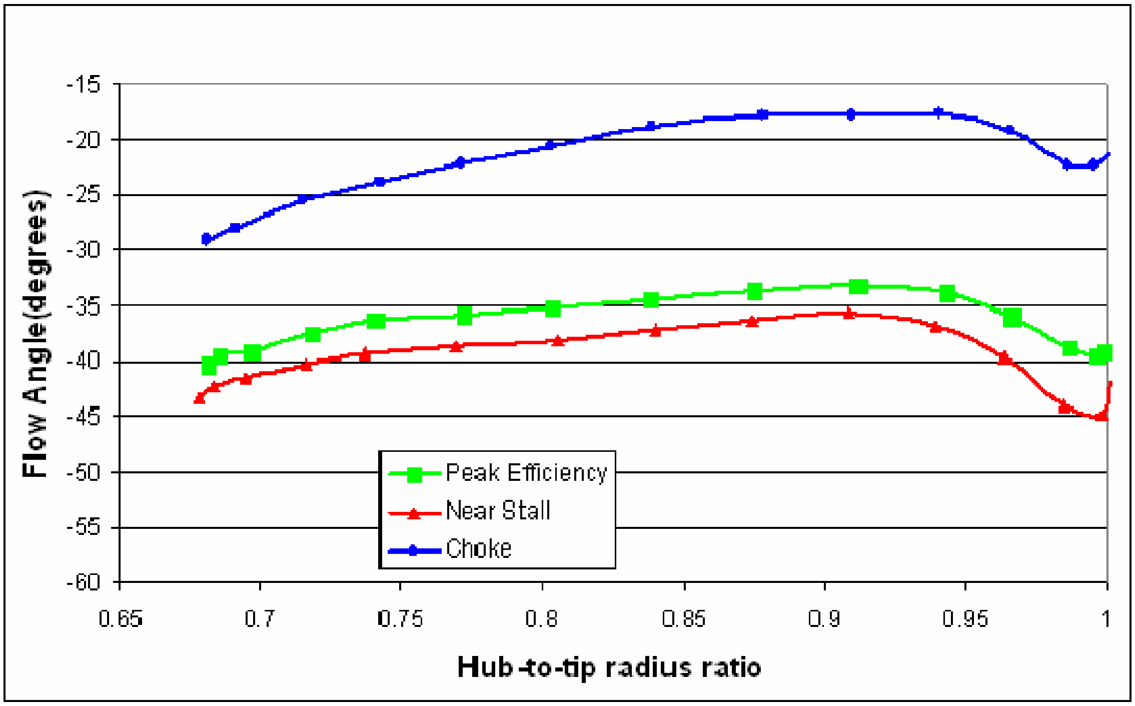


Figure 39. Exit flow angle, 100% speed.

3. Diffusion Factor

The diffusion factor is a method of assessing the blade loading in an axial compressor. For a simple, two-dimensional geometry, the diffusion factor was calculated using the equation:

$$DF := 1 - \frac{V_2}{V_1} + \frac{V_{\theta 2}}{2 \cdot \sigma \cdot V_1}$$

Where V_1 and V_2 were the inlet and exit velocities relative to the blade, $V_{\theta 2}$ was the turning of the flow, assuming the flow is axial at the blade entrance, and σ is the solidity of the blade corresponding to the flow. It was believed that diffusion factor values above 0.6 could indicate blade stall and values of 0.45 may be a typical design choice. (Cumpsty, 1989)

The relative inlet velocity, V_1 was calculated using the velocity of the rotor and by taking the area average of the velocities found upstream and calculating the flow at the leading edge of the rotor assuming the decreasing area acts as a nozzle. The relative exit velocity, V_2, was calculated using velocity of the rotor from the downstream probe angle and velocity measurement and calculating the radial and axial components of flow. From these components, the relative exit velocity was determined.

The solidity was calculated using data from the rotor blade that had been used in the SWIFT calculations. A graph of the solidity can be found in Appendix O.

The diffusion factor was calculated for each set of speed data. A sample set of calculations for 100% Speed, Peak Efficiency, near the midpoint of the blade is shown in Appendix O. Three-dimensional representations of the diffusion factor are shown for Choke, Peak Efficiency and Stall in Figures 40-42.

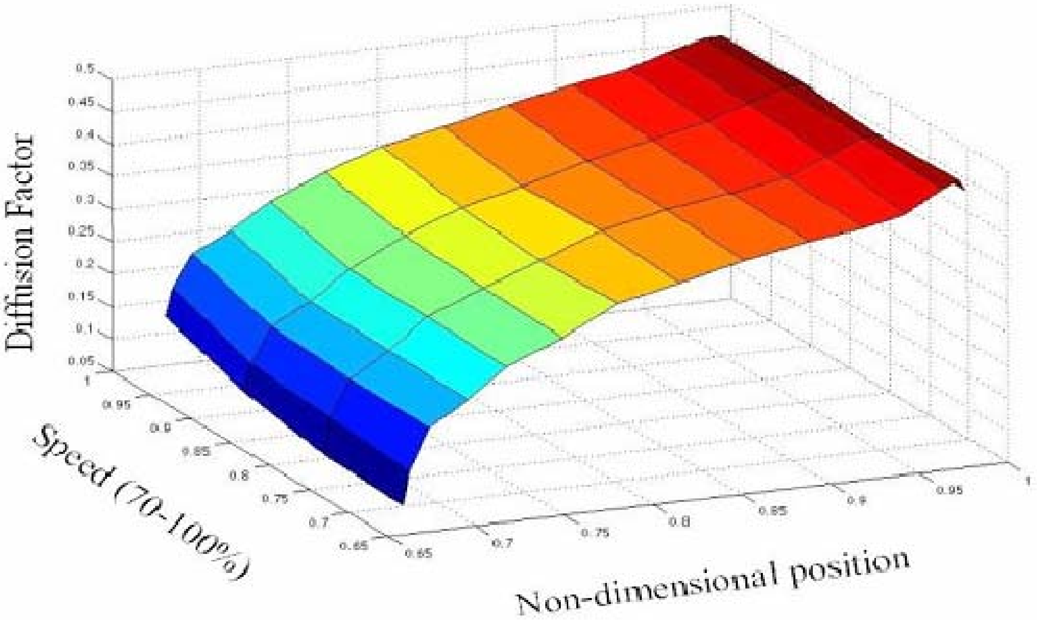


Figure 40. Diffusion Factor at Choke.

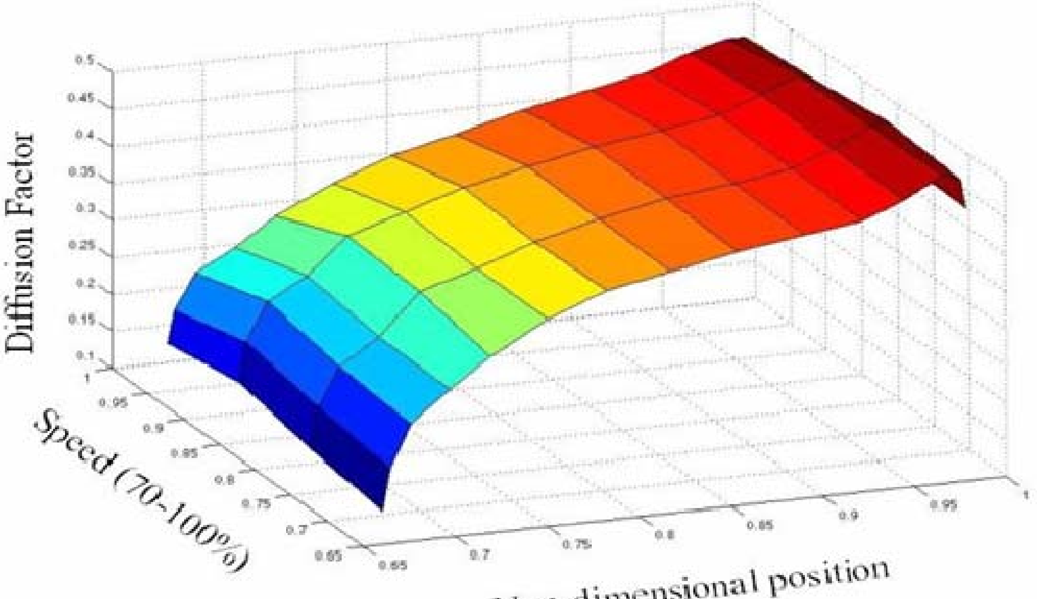


Figure 41. Diffusion Factor at Peak Efficiency.

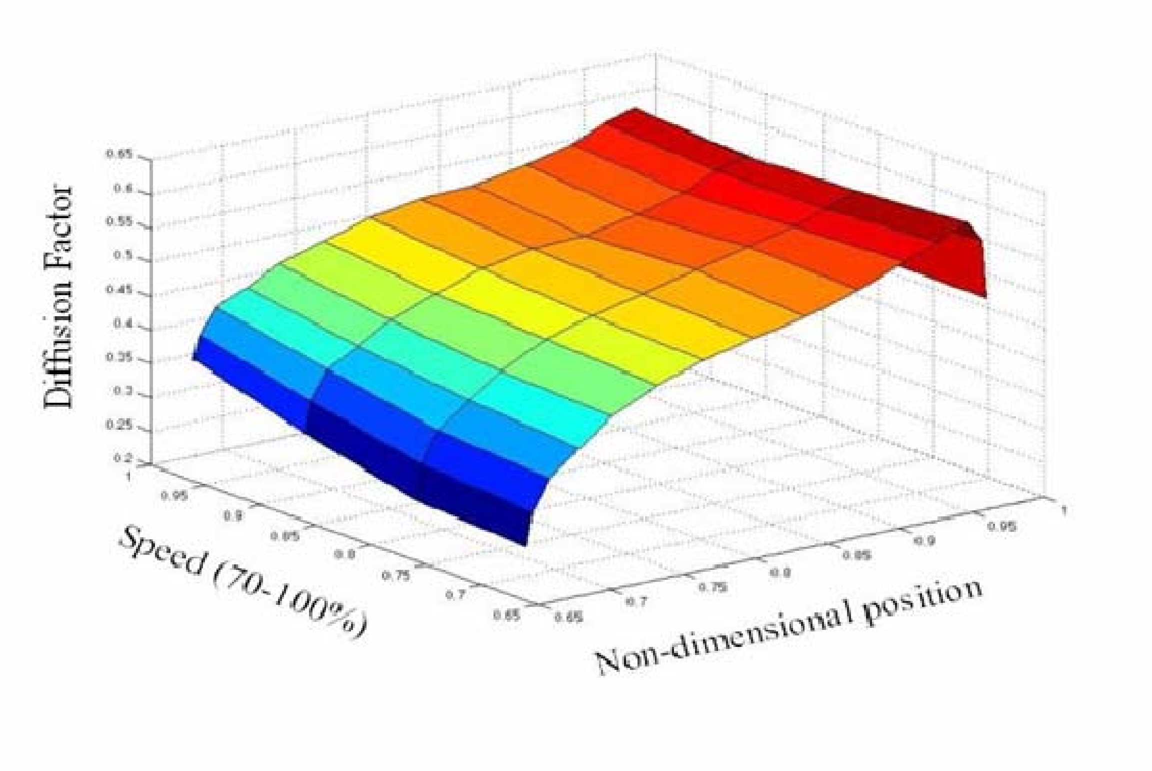


Figure 42. Diffusion Factor at Near Stall.

B. OBSERVATION

1. Tip Stall

The experimental data predicted a tip stall as the mass flow rate is decreased. In Figures 40-42, the diffusion factor peaks near the tip. Some research has been conducted on the shock structure and location of the tip leakage vortex originating at the location of the maximum pressure differential at the tip of the blade. (Gannon et al., 2004)

2. Magnitude of Diffusion Factor

As the speed is increased or the mass flow rate is decreased, the diffusion factor increased. This demonstrated the increased blade loading for those conditions. The general value for the diffusion factor of an axial compressor is 0.45 and the value for the diffusion factor at stall is 0.6. (Cumpsty, 1989) In Figure 42, which shows the diffusion factor at near stall for speeds 70-100%, the diffusion factor approached 0.6 at the tip for all speeds measured. The high diffusion factors at the tip indicated that the rotor would undergo tip stall as was tested and described in "Measurement of the Unsteady Case Wall Pressures." (Gannon, Hobson, Shreeve, 2005)

3. Comparison of Exit Plane CFD Data to Experimental Data

The CFD data produced by the SWIFT code was compared to experimental data taken from downstream of the rotor for the equivalent pressure ratios. The following graphs demonstrate the high level of accuracy of the CFD for the 90% speed. At peak efficiency, the CFD total pressure ratio (Figure 43) exhibited nearly the same form and produced data very close to that of the experimental. Both Mach number calculations (Figure 44) and flow angles (Figure 45) were also very similar and all three graphs are shown below.

However, an inaccuracy was introduced during the production of the CFD data. In the experimental model, the hub did not rotate downstream of the rotor. In the CFD grid, a rotating hub was used in the model downstream of the rotor, producing discrepancies in the flow close to the hub. This is more evident in the flow angle graph shown in Figure 45. Additional graphs for all settings at 90% speed can be found in appendix M.

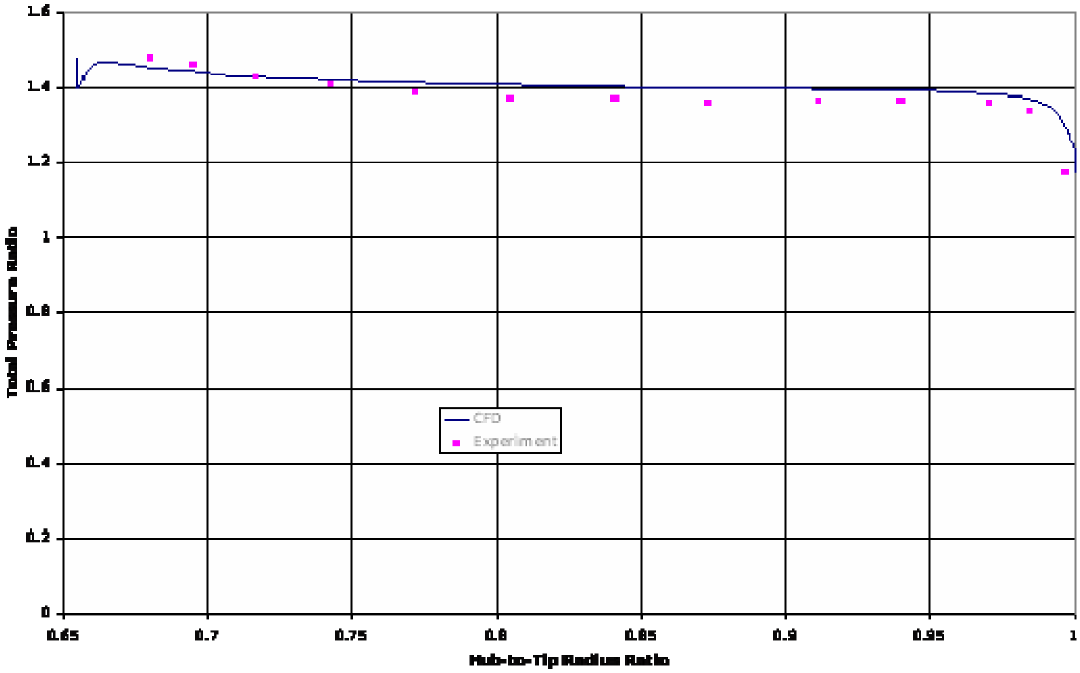


Figure 43. CFD vs experimental data, 90% speed, peak efficiency: total pressure ratio.

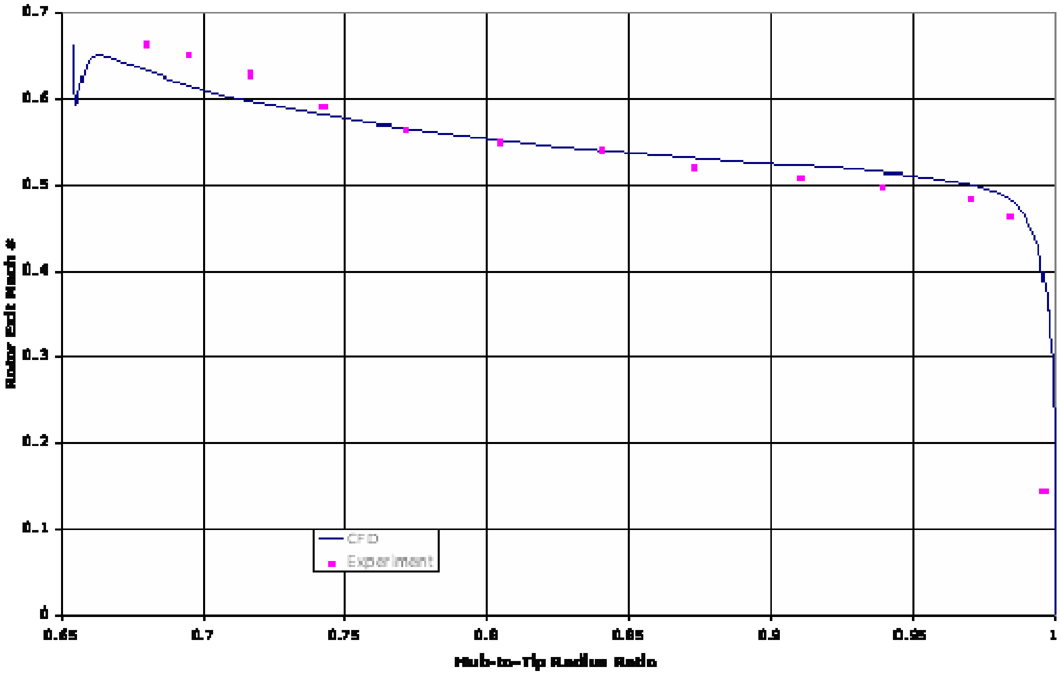


Figure 44. CFD vs experimental data, 90% speed, peak efficiency: Mach number.

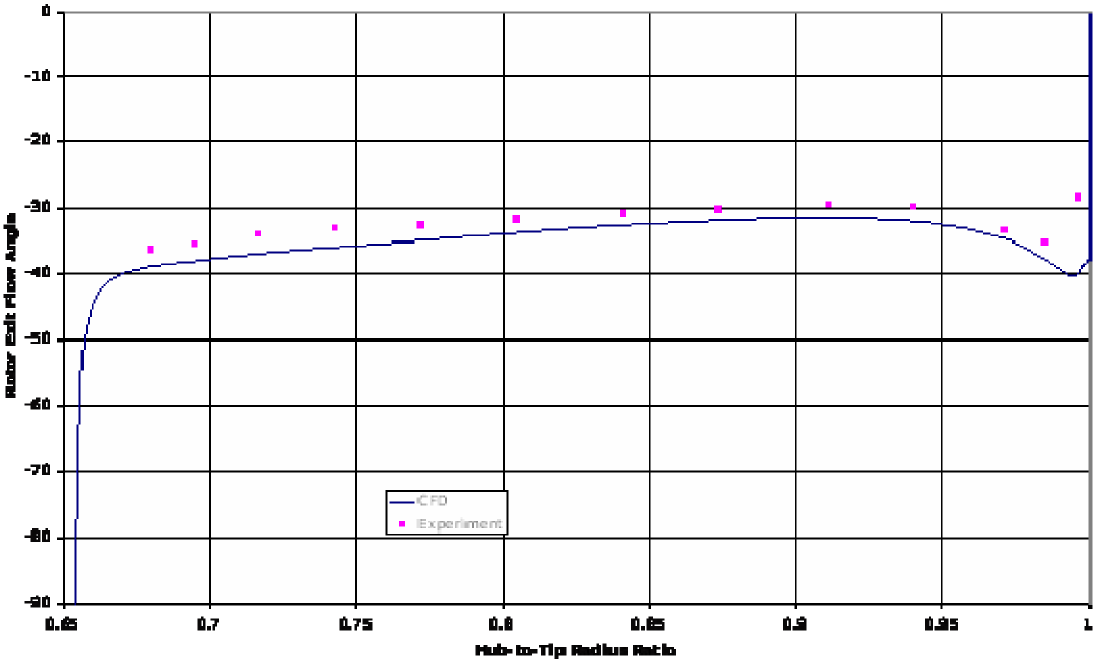


Figure 45. CFD vs experimental data, 90% speed, peak efficiency: flow angle.

4. Comparison of CFD Diffusion Factor Data to Experimental Results

The diffusion factor results were graphed and compared to the SWIFT code results. The CFD data also predicted a tip stall. Along the outer half of the blade radius, the CFD and experimental data exhibited the same characteristics and obtained very close results for both near stall and peak efficiency. The choke data, however, showed the largest discrepancy. In addition, in the inner half of the blade radius, the experimental and CFD data exhibited different characteristics, although the magnitude of the data was similar. This also can be attributed to the rotating hub used in the geometric grid in the CFD calculations. Traveling radially inward, the experimental data decreased towards zero, while the CFD data decreased by the same magnitude and then increased. The results are shown in Figures 46-49.

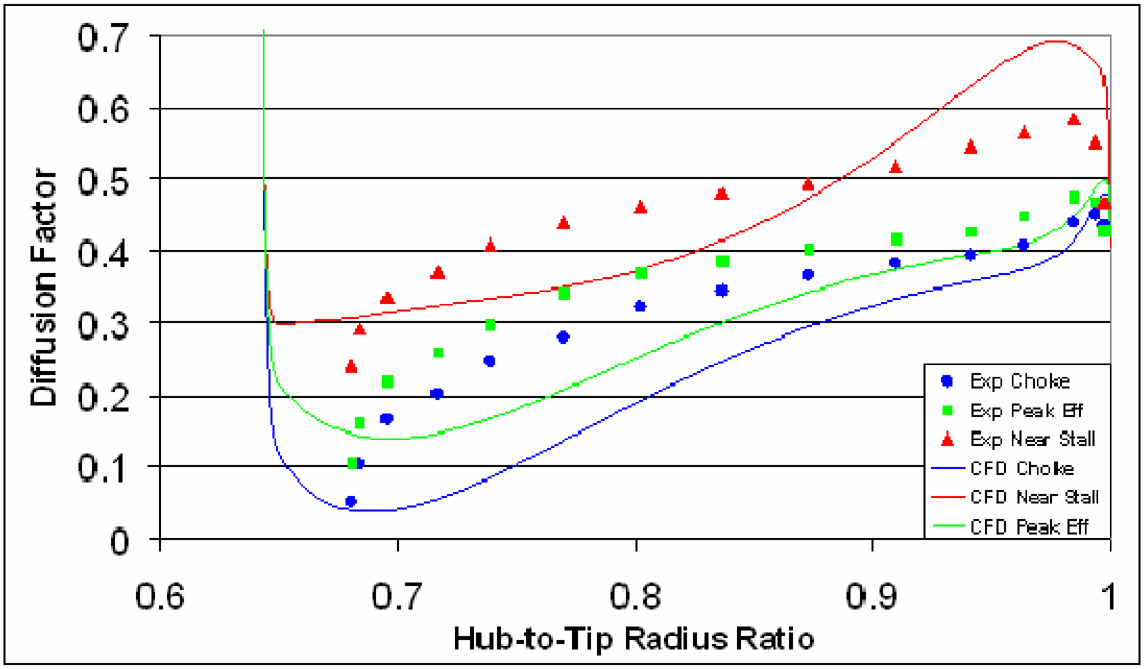


Figure 46. 70% speed diffusion factor comparison.

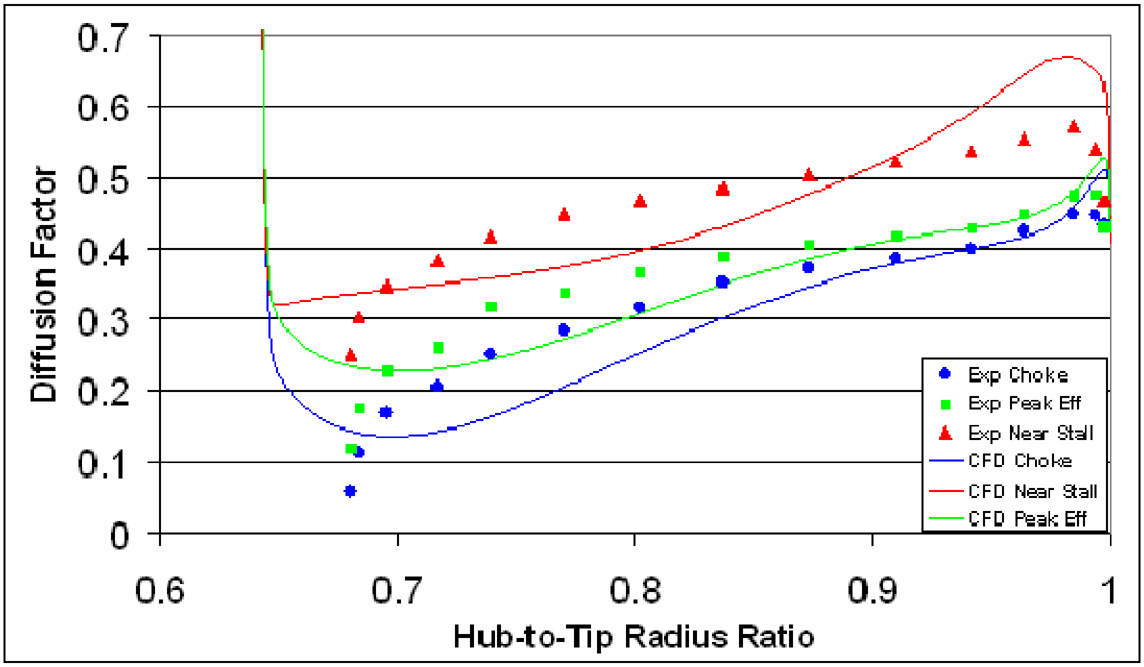


Figure 47. 80% speed diffusion factor comparison.

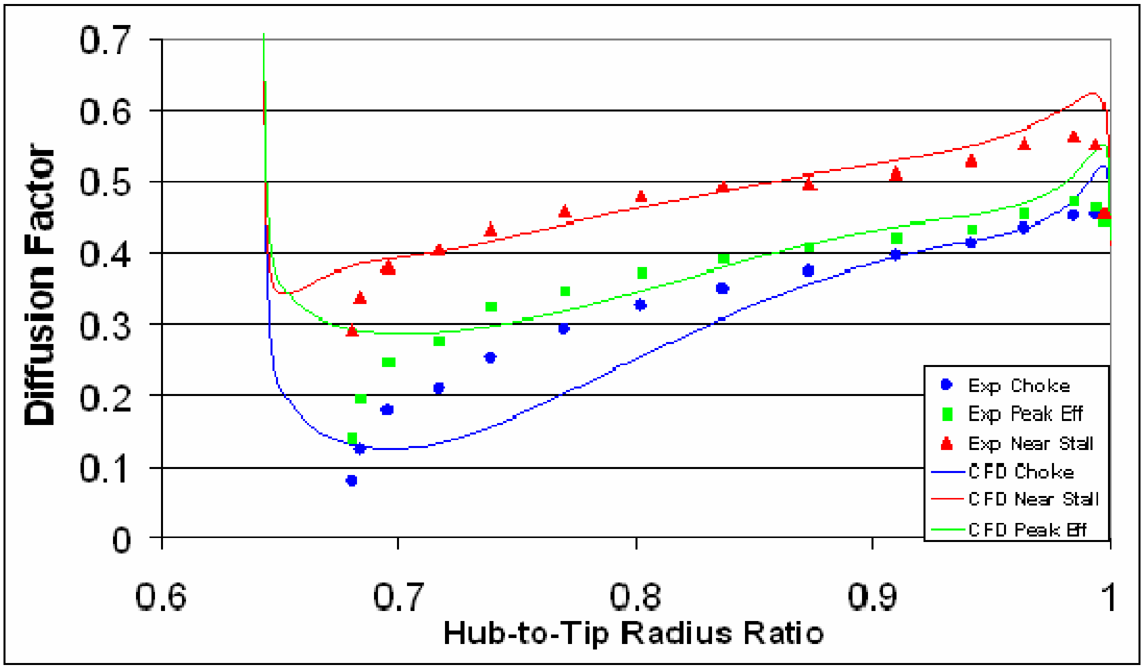


Figure 48. 90% speed diffusion factor comparison.

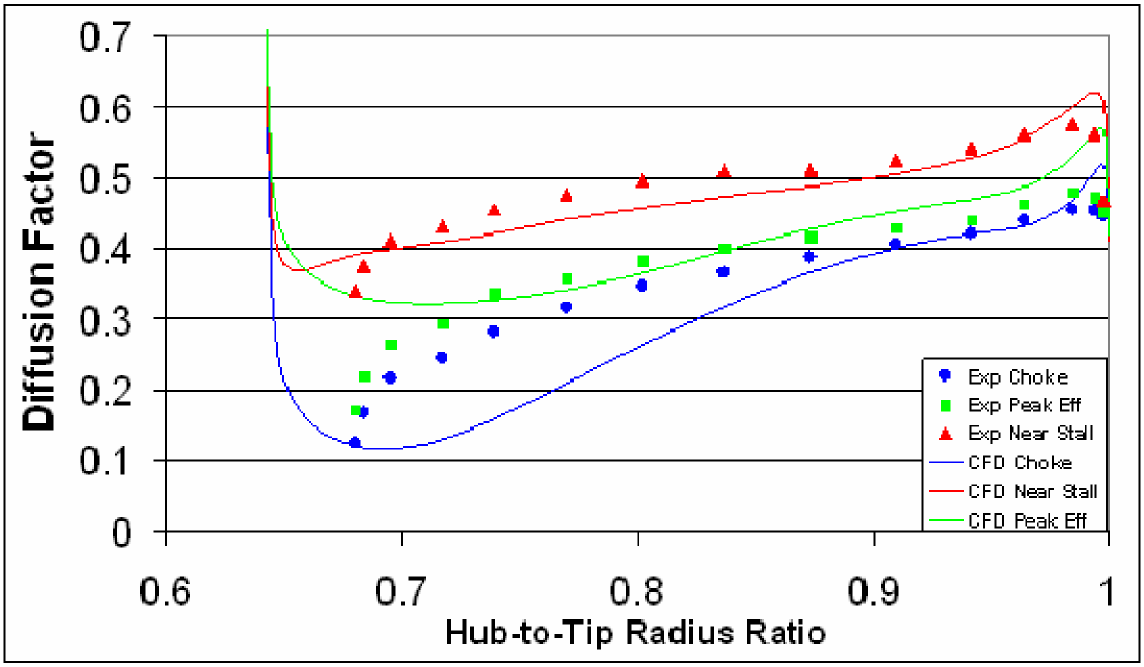


Figure 49. 100% speed diffusion factor comparison.

THIS PAGE INTENTIONALLY LEFT BLANK

VII. CONCLUSIONS AND RECOMMENDATIONS

A. CONCLUSIONS

The present study set up the instrumentation and established a baseline for future investigation into stall and surge in a transonic axial compressor. The "pop stall" phenomenon experienced by jet aircraft ingesting hot gas was the primary purpose of the present work.

The initial probe calibration was conducted by manually rotating the probe with an electric switch controller. Rotating the probe to a specified angle using the manual controller was laborious and time consuming.

The final system of instrumentation created to evaluate flow upstream and downstream of the compressor reduced the amount of work and time required. The Matlab control system using USB devices worked rapidly, decreasing the time required for probe movements. The pressure transducers provided stable, accurate data to a system capable of processing the data as it was taken and were capable of being calibrated to a lower Mach number. The USB A/D converter allowed rapid communication with Matlab and allowed the fast processing of data during the experiment. In addition, the USB A/D converter provided feedback for the probe actuator as well as the processing of pressure data to allow feedback for to null yaw the probe. However, the Matlab programming could be optimized as the system slowed occasionally and failed when the probe was placed in a highly swirling flow.

The nature of the diffusion factor calculated was found to be close to what was predicted by CFD code and the magnitude was within the expected range. In addition, the shape of the diffusion factor data shows the higher loading at the tip and validates the prediction of a tip stall.

B. RECOMMENDATIONS

All data was collected with a single, three-hole probe. To calculate the diffusion factor, data from both upstream and downstream was necessary. The data was collected on separate days, with different atmospheric conditions. Compressor speed and mass flow rates were repeated, giving close approximate reproductions of flow conditions. To ensure less error, the use of two probes, giving simultaneous data upstream and downstream is recommended.

Data downstream of the rotor was collected with a three-hole probe that was not sensitive to pitch. CFD models have shown that the flow at the exit of the rotor contains significant radial components, as well as axial and tangential velocity components. A five-hole probe would be better able to determine both pitch and yaw angles and give a better representation of the downstream flow.

The temperature data used in this experiment was provided by thermocouples connected to the HPVEE system run from a separate PC. More accurate data collection would be possible with a thermocouple connected to a USB device controlled with Matlab. Temperature measurements would then be collected simultaneously with the pressure data.

The diffusion factor calculations require the relative entrance velocity and relative exit velocity of the flow referenced to the blade. The diffusion factor was calculated from holes more than four inches upstream and two inches downstream of the rotor. Gathering data closer to the rotor would increase the accuracy of the diffusion factor calculations.

The Matlab functions used to control the probe and sample data could be optimized. A better logic loop for rotating to an angle or traversing to a position could reduce the time required for each experiment. In addition, the calling of calibration constants required a large amount of computer memory to load the constants in the MAT file and interpolate the angle and Mach number and may require even more memory for a five-hole probe.

LIST OF REFERENCES

1. Chima, R.V., 1998, “Calculation of Multistage Turbomachinery Using Steady Characteristics Boundary Layer Conditions,” AIAA-98-0968 or NASA/TM-1998-206613.

2. Cumpsty, N.A., Compressor Aerodynamics, 2nd Edition, 1989, John Wiley & Sons, New York, New York.

3. Donelson, S. Briggs, T., "*JSF Team Probes Steam Catapult Environment*," JSF Integrated Test Force, dcmilitary.com, February 6, 2003.

4. Gannon, A.J., Hobson, G.V. and Shreeve, R.P., 2004, "A transonic Compressor Stage Part 1: Experimental Results," ASME GT2004-53923, Turbo Expo, Vienna Austria.

5. Gannon, A.J., Hobson, G.V., Shreeve, R.P., “Measurement of the Unsteady Casewall Pressures Over the Rotor of a Transonic Fan and Comparison with Numerical Predictions", ISABE 2005, 17th International Symposium on Airbreathing Engines, Munich, September 2005.

6. Greco, P.A., Turbine Performance Mapping of the Space-Shuttle Main Engine High-Pressure Fuel Turbopump, Master's Thesis, Naval Postgraduate School, Monterey, California, September 1995.

7. Grossman, B.L., Testing and Analysis of a Transonic Axial Compressor, Master's Thesis, Naval Postgraduate School, Monterey California, September 1997.

8. Hobson, G.V, Gannon, A.J., and Shreeve, R.P., 2004, "A transonic Compressor Stage Part 2: CFD Simulation," ASME GT2004-53927, Turbo Expo, Vienna Austria.

9. Johansen, E.S., Rediniotis, O.K., Jones, G., "The Compressible Calibration of Miniature Multi-Hole Probes," Transactions of the ASME, March 2001, Vol 123, pp 128-138.

10. Measurement Computing, PMD-1608FS USB Based Analog and Digital I/O Module User's Guide. May 6, 2005.

11. Measurement Computing, USB-ERB24 USB Based 24 Relay Module User's Guide. May 4, 2005.

12. O'Brien, J.M., Transonic Compressor Test Rig Rebuild and Initial Results with the Sanger Stage, Master's Thesis, Naval Postgraduate School, Monterey, California, June 2000.

13. Omega.com, "Low Cost Silicon Pressure Sensor, PX 138 Series", 2005, Stamford Connecticut.

14. Papamarkos, I., Inlet Distortion Generation for a Transonic Compressor, Master's Thesis, Naval Postgraduate School, Monterey, California, September 2004.

15. Reid, William D., Transonic Axial Compressor Design Case Study and Preparations for Testing, Master's Thesis, Naval Postgraduate School, Monterey, California, September 1995.

16. Roberts, M.W. Unsteady Pressure Measurements on the Case Wall of a Transonic Compressor, Master's Thesis, Naval Postgraduate School, Monterey, California, June 2003.

17. Sanger, N. L., "Design of a Low Aspect Ratio Transonic Compressor Stage Using CFD Techniques," ASME Journal of Turbomachinery, July 1996, Vol. 118 pp 479-491.

18. Sanger, N. L., "*Design Methodology for the NPS Transonic Compressor*," TPL Technical Note 99-01, August 1999.

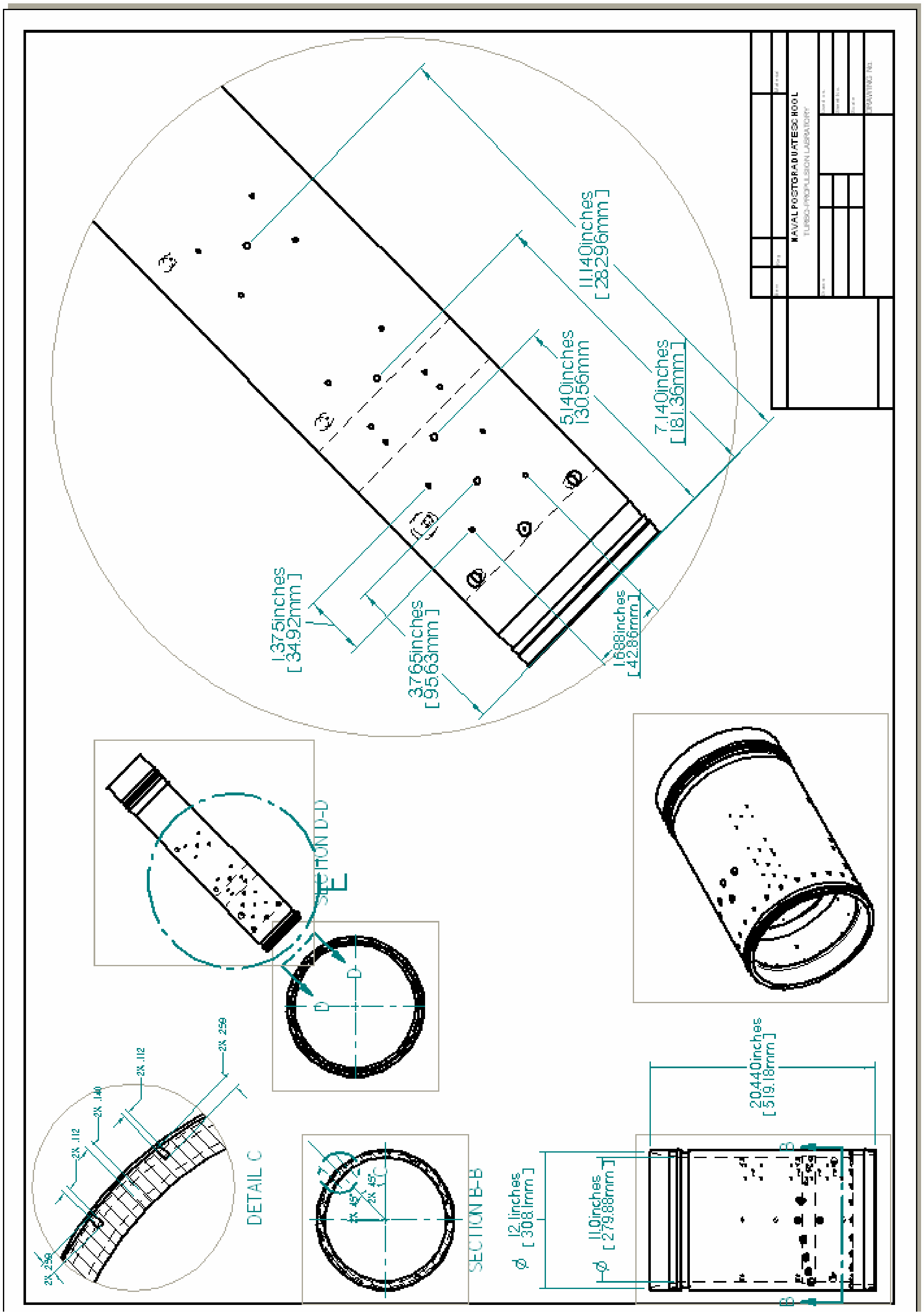
19. Sussman Automatic Corporation, "SVS6 Electric Steam Boilers Specifications.", 2004, Long Island City, New York.

20. The MathWorks Inc., "Using Matlab Graphics," Release 13, Version 6.5, September 2002.

21. White, F.M., Fluid Mechanics, 5th Edition, 2003, McGraw-Hill, New York, New York.

THIS PAGE INTENTIONALLY LEFT BLANK

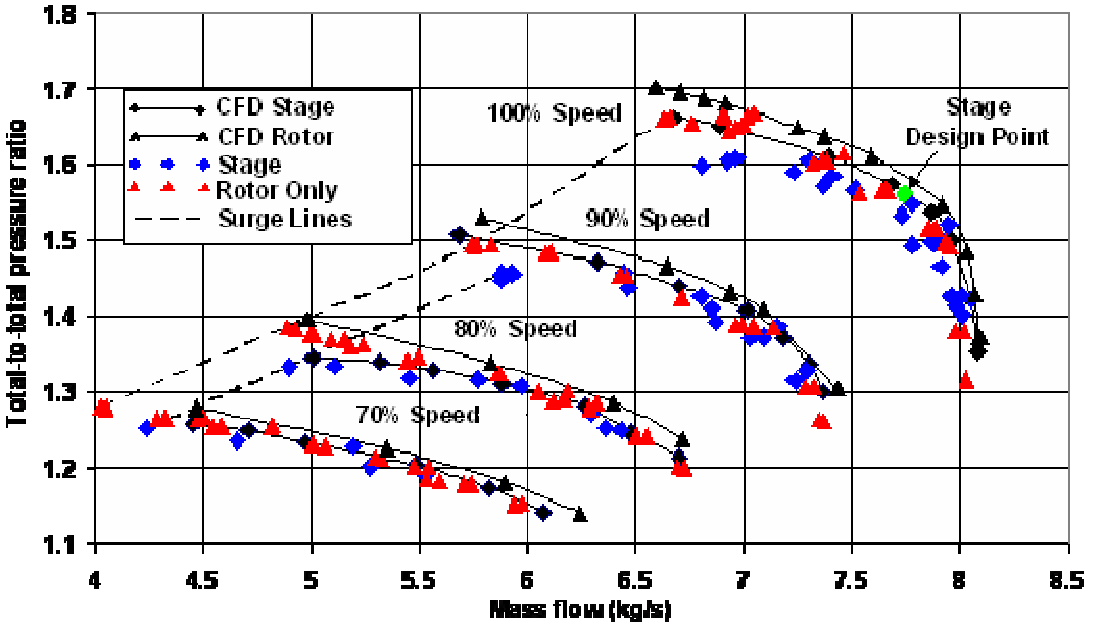
APPENDIX A: CASE WALL ENGINEERING DRAWING

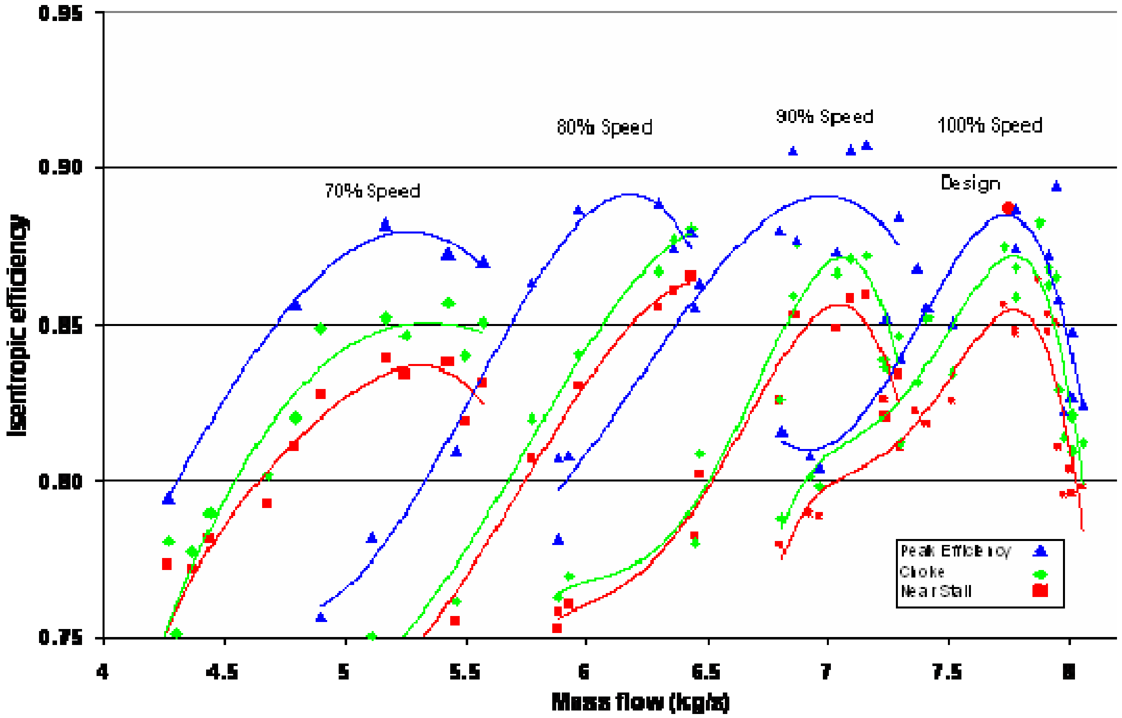


Probe Hole Distances from Trailing and Leading Edges of Rotor

Hole Number	Hole Location
1	111.5mm (4.39") Upstream of Rotor Leading Edge
2	9.91mm (.39") Upstream of Rotor Leading Edge
3	8.64mm (.34") Downstream of Rotor Trailing Edge
4	59.4mm (2.34") Downstream of Rotor Trailing Edge

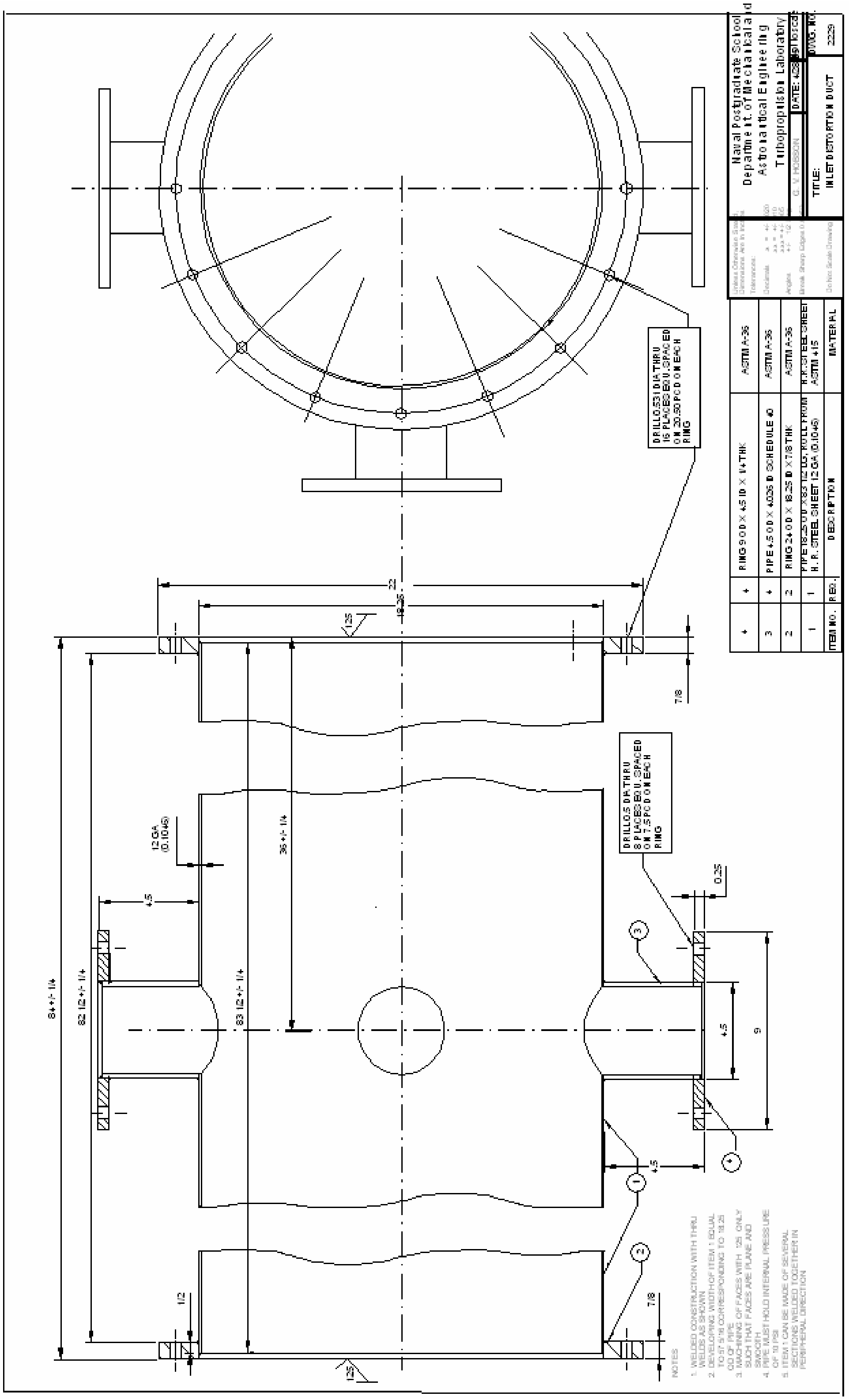
APPENDIX B: COMPLETE STAGE COMPRESSOR MAP





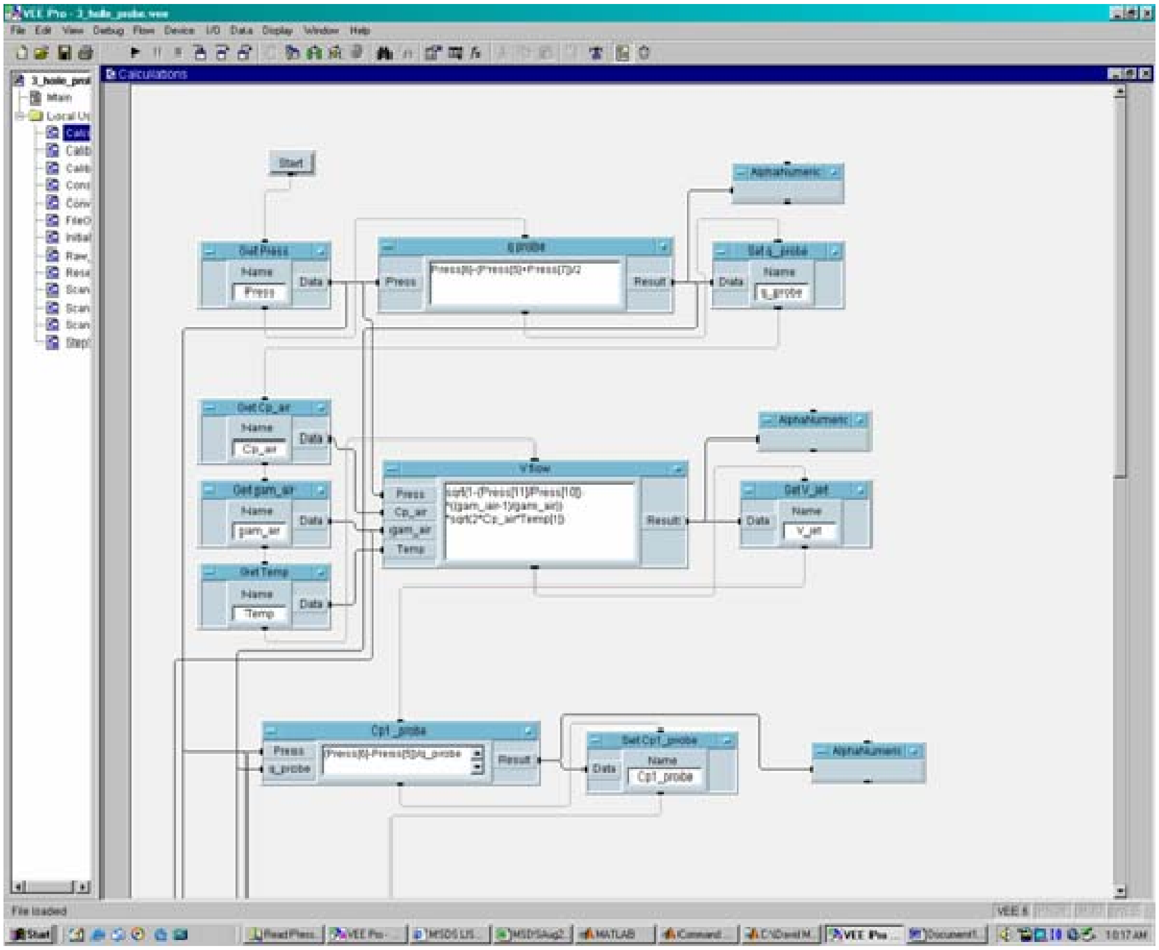
THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX C: COMPRESSOR INTAKE SYSTEM


Naval Postgraduate School
Department of Mechanical and
Astronautical Engineering
Turbopropulsion Laboratory
G. V. HOBSON
DATE: 4/28/99
DWG. NO.
2229
TITLE:
INLET DISTORTION DUCT
Unless Otherwise Stated, Dimensions Are in Inches
Tolerances:
Decimals
Angles
Break Sharp Edges
Do Not Scale Drawing
4 | 4 | RING 9 O D X 4.5 ID X 1/4 THK | ASTM A-36
3 | 4 | PIPE 4.5 O D X 4.026 ID SCHEDULE 40 | ASTM A-36
2 | 2 | RING 24 O D X 18.25 ID X 7/8 THK | ASTM A-36
1 | 1 | PIPE 18.25 OD X 83 1/2 LG, ROLL FROM H. R. STEEL SHEET 12 GA (0.1046) | H.R. STEEL SHEET ASTM A15
ITEM NO. | REQ. | DESCRIPTION | MATERIAL
DRILL 0.531 DIA THRU 16 PLACES EQU. SPACED ON 20.50 PCD ON EACH RING
DRILL 0.5 DIA THRU 8 PLACES EQU. SPACED ON 7.5 PCD ON EACH RING
22
18.25
125
7/8
12 GA (0.1046)
4.5
36 +/- 1/4
84 +/- 1/4
82 1/2 +/- 1/4
83 1/2 +/- 1/4
0.25
4.5
9
1/2
3
4
1
2
NOTES
1. WELDED CONSTRUCTION WITH THRU WELDS AS SHOWN
2. DEVELOPING WIDTH OF ITEM 1 EQUAL TO 57 5/16 CORRESPONDING TO 18.25 OD OF PIPE
3. MACHINING OF FACES WITH 125 ONLY SUCH THAT FACES ARE PLANE AND SMOOTH
4. PIPE MUST HOLD INTERNAL PRESSURE OF 10 PSI
5. ITEM 1 CAN BE MADE OF SEVERAL SECTIONS WELDED TOGETHER IN PERIPHERAL DIRECTION

THIS PAGE INTENTIONALLY LEFT BLANK

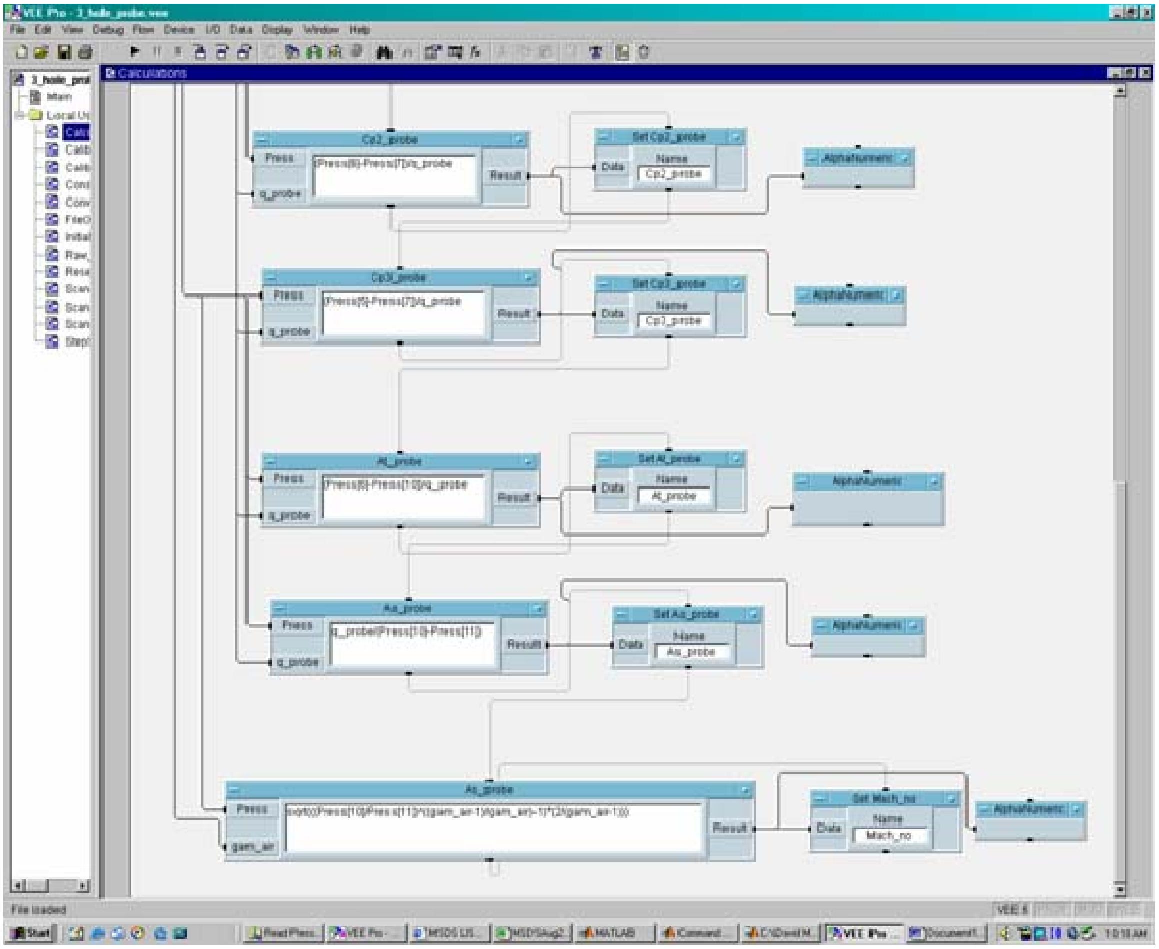
APPENDIX D: SCREEN SHOTS OF THE HPVEE PROGRAM



```
Press[6]-(Press[5]+Press[7])/2
```

```
sqrt(1-(Press[11]/Press[10])
^((gam_air-1)/gam_air))
*sqrt(2*Cp_air*Temp[1])
```

```
(Press[6]-Press[5])/q_probe
```



Cp2_probe
Press
q_probe
(Press[6]-Press[7])/q_probe
Result

Set Cp2_probe
Data
Name: Cp2_probe

AlphaNumeric

Cp3_probe
Press
q_probe
(Press[5]-Press[7])/q_probe
Result

Set Cp3_probe
Data
Name: Cp3_probe

AlphaNumeric

At_probe
Press
q_probe
(Press[6]-Press[10])/q_probe
Result

Set At_probe
Data
Name: At_probe

AlphaNumeric

As_probe
Press
q_probe
q_probe/(Press[10]-Press[11])
Result

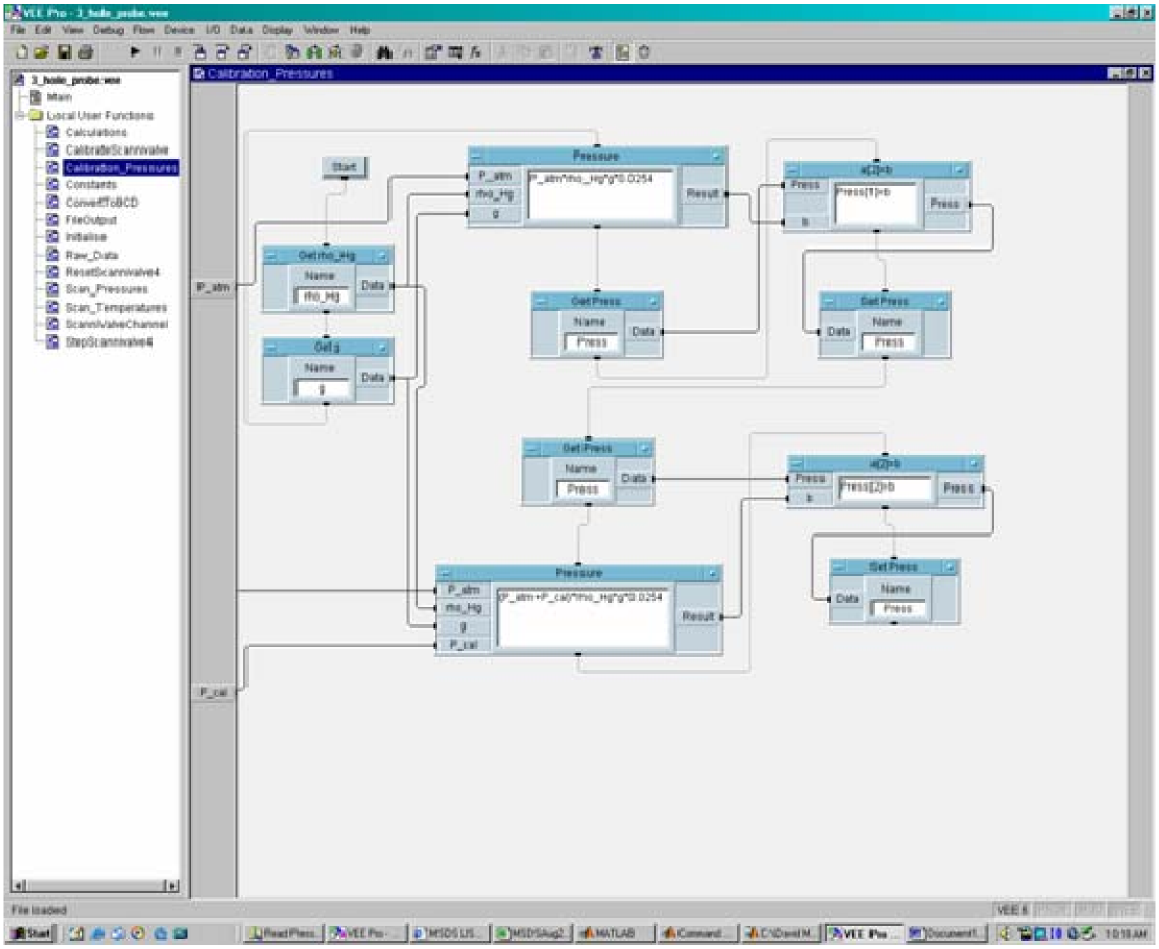
Set As_probe
Data
Name: As_probe

AlphaNumeric

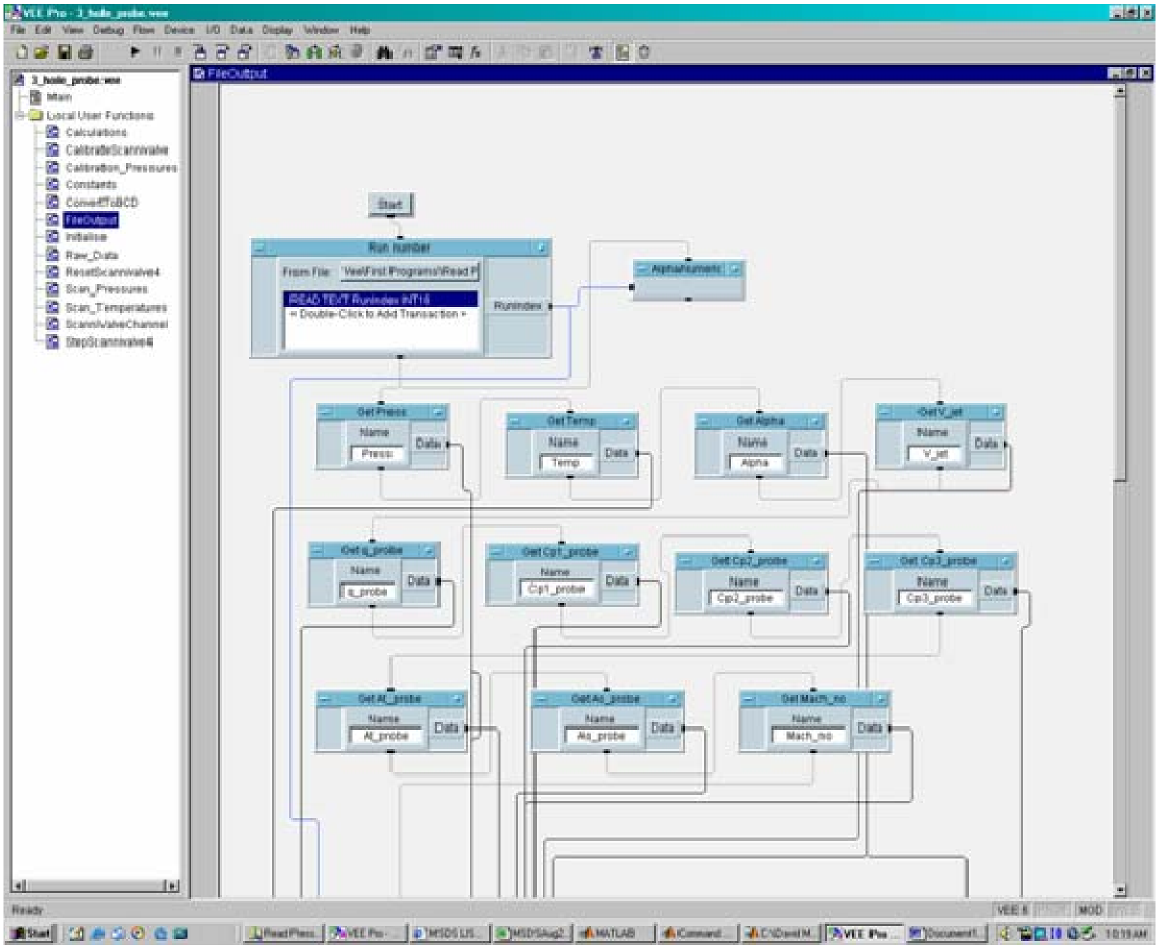
As_probe
Press
gam_air
sqrt(((Press[10]/Press[11])^((gam_air-1)/gam_air)-1)*(2/(gam_air-1)))
Result

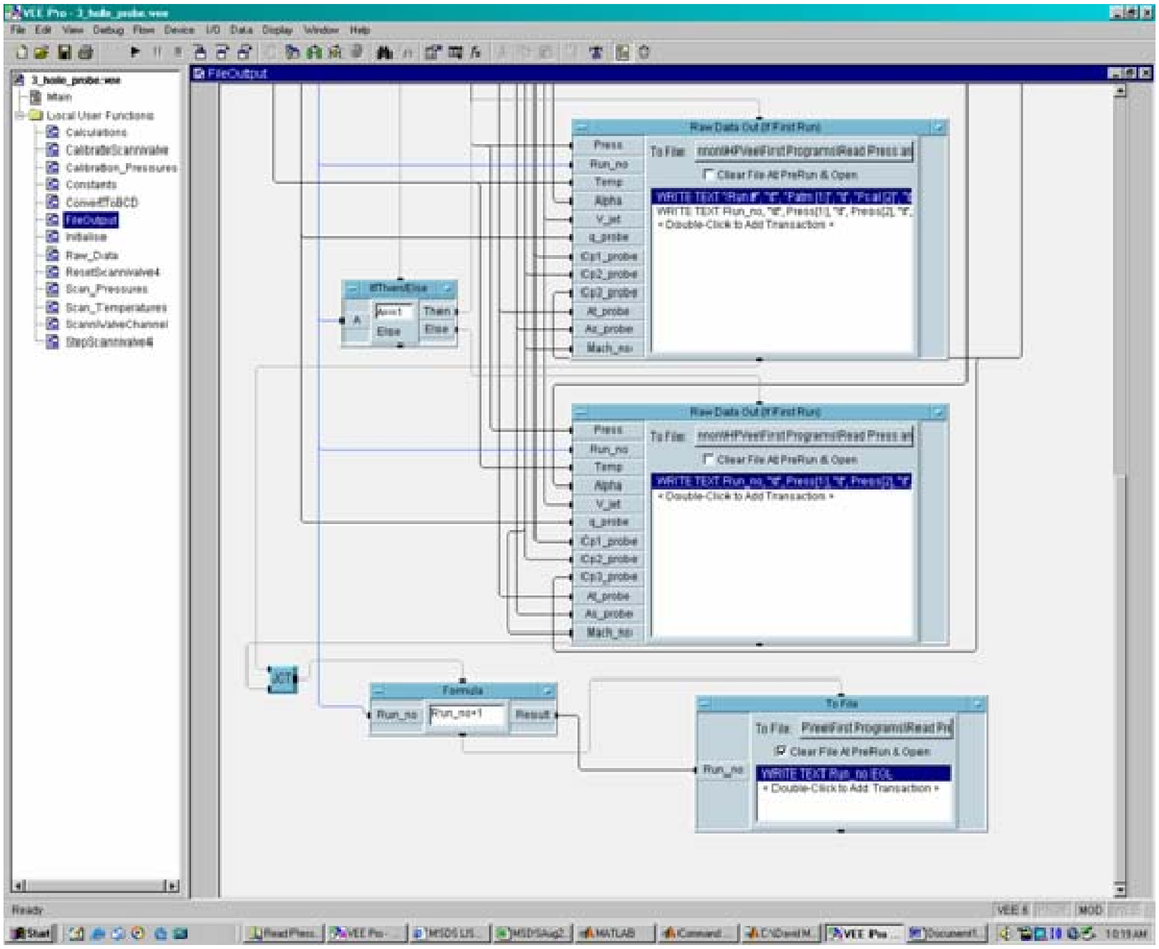
Set Mach_no
Data
Name: Mach_no

AlphaNumeric



```
P_atm*rho_Hg*g*0.0254
Press[1]=b
Press[2]=b
(P_atm+P_cal)*rho_Hg*g*0.0254
```


VEE Pro - 3_hole_probe.vee
File Edit View Debug Flow Device I/O Data Display Window Help
3_hole_probe.vee
Main
Local User Functions
Calculations
CalibrateScannivalve
Calibration_Pressures
Constants
ConvertToBCD
FileOutput
Initialise
Raw_Data
ResetScannivalve4
Scan_Pressures
Scan_Temperatures
ScannivalveChannel
StepScannivalve4
FileOutput
Start
Run number
From File: Vee\First Programs\Read P
READ TEXT RunIndex INT16
« Double-Click to Add Transaction »
RunIndex
AlphaNumeric
Get Press
Name
Press
Data
Get Temp
Name
Temp
Data
Get Alpha
Name
Alpha
Data
Get V_jet
Name
V_jet
Data
Get q_probe
Name
q_probe
Data
Get Cp1_probe
Name
Cp1_probe
Data
Get Cp2_probe
Name
Cp2_probe
Data
Get Cp3_probe
Name
Cp3_probe
Data
Get At_probe
Name
At_probe
Data
Get As_probe
Name
As_probe
Data
Get Mach_no
Name
Mach_no
Data
Ready
VEE 6
MOD
Start
Read Press...
VEE Pro - ...
MSDS LIS...
MSDSAug2...
MATLAB
Command...
C:\David M...
VEE Pro ...
Document1...
10:19 AM



Else: A==1

To File: nnon\HP\Vee\First Programs\Read Press an

□ Clear File At PreRun & Open

WRITE TEXT "Run #", "\t", "Patm [1]", "\t", "Pcal [2]", "\t

WRITE TEXT Run_no, "\t", Press[1], "\t", Press[2], "\t",

< Double-Click to Add Transaction >

To File: nnon\HP\Vee\First Programs\Read Press an

□ Clear File At PreRun & Open

WRITE TEXT Run_no, "\t", Press[1], "\t", Press[2], "\t"

< Double-Click to Add Transaction >

Run_no+1

To File: P\vee\First Programs\Read Pr

☑ Clear File At PreRun & Open

WRITE TEXT Run_no EOL

< Double-Click to Add Transaction >

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX E: CALIBRATION EQUATIONS AND SAMPLE CALCULATION

In the calibration, β is the yaw angle taken from the angle of the probe actuator, which was set to -20° with a .1° tolerance.

$$\beta := -19.928$$

The pressures P_1-P_3 were taken from the three-hole probe.

$$P_1 := 117780.4\text{Pa}$$

$$P_2 := 117318.3\text{Pa}$$

$$P_3 := 101240.6\text{Pa}$$

P_{oj} and P_{sj} were taken from the Stagnation and Static Pressure Probes.

$$P_{oj} := 120177\text{Pa}$$

$$P_{sj} := 100783\text{Pa}$$

$$T_p := 297.5\mathrm{K}$$

$$C_p := 1005\frac{\mathrm{J}}{\mathrm{kg}\cdot\mathrm{K}}$$

$$\gamma := 1.41$$

The pseudo dynamic pressure, q_probe is given as:

$$\text{q\_probe} := P_1 - \frac{\left(P_2 + P_3\right)}{2}$$

$$\text{q\_probe} = 8.501 \times 10^3\,\mathrm{Pa}$$

$$\text{V\_jet} := \sqrt{\left[1 - \left(\frac{P_{sj}}{P_{oj}}\right)^{\frac{(\gamma-1)}{\gamma}}\right]}\cdot\sqrt{2\cdot C_p\cdot T_p}$$

$$\text{V\_jet} = 172.72\frac{\mathrm{m}}{\mathrm{s}}$$

The angle flow discriminators give the direction of the flow.

$$C_{p\_1} := \frac{\left(P_1 - P_2\right)}{q\_probe}$$

$$C_{p\_1} = 0.054$$

$$C_{p\_2} := \frac{\left(P_1 - P_3\right)}{q\_probe}$$

$$C_{p\_2} = 1.946$$

$$C_{p\_3} := \frac{\left(P_2 - P_3\right)}{q\_probe}$$

$$C_{p\_3} = 1.891$$

The total and static pressure coefficients provide the measurable data for discerning unknown flows.

$$A_{t\_Probe} := \frac{\left(P_1 - P_{oj}\right)}{q\_probe}$$

$$A_{t\_Probe} = -0.282$$

$$A_{s\_Probe} := \frac{q\_probe}{\left(P_{oj} - P_{sj}\right)}$$

$$A_{s\_Probe} = 0.438$$

$$MachNo := \sqrt{\left[\left(\frac{P_{oj}}{P_{sj}}\right)^{\frac{(\gamma-1)}{\gamma}} - 1\right]\cdot\left(\frac{2}{\gamma - 1}\right)}$$

$$MachNo = 0.506$$

APPENDIX F: MATLAB CALIBRATION FILES

```
function[] = calibration_run(angle1, angle2, points, Tp, Tj)

positions=linspace(angle1, angle2, points);



for i=1:points

    Input_Angle=positions(i);


    handles = guihandles(gcbo); % generate handles struct

    set(handles.Input_Angle, 'String', Input_Angle);

    des_angle=Input_Angle;

    rotate(des_angle);

    pause(2);

    OUTPUT_TO_EXCEL(Tp, Tj);

end



-------------------------------------------------------------------------------------------------
function[cur_angle] = rotate(des_angle)

%this function calls the data from the GUI, PMD_1608FS, and tells the
%ERB_24 to rotate the probe actuator.  The desired angle (des_angle) is input from the
GUI.


%For Loop to get to the proper angle
for i=1:15;

    %Get the current analog angle and call it 'h'.
    h=PMD_1608FS_2;

    %Convert the analog angle into a real angle.
    cur_angle=angle_calibration(h);


    handles = guihandles(gcbo); % generate handles struct
```

```
   %Set the value of the Cur_Angle textbox to the current angle.
   set(handles.Cur_Angle, 'String', cur_angle);

   %Assign the cur_angle to be used in the base workspace.
   assignin('base','cur_angle',cur_angle)

   %Find the difference between the current angle and the required angle.
   diff=cur_angle-des_angle;
   if abs(diff)<=.1
      break;
   end



   if cur_angle>=des_angle&&abs(diff)>=20

      ERB24_LEFT(6);
   pause(.1)
elseif cur_angle>=des_angle&&abs(diff)<=19.99999&&abs(diff)>=10

   ERB24_LEFT(3);
   pause(.1)
elseif cur_angle>=des_angle&&abs(diff)<=9.99999&&abs(diff)>=5
   ERB24_LEFT(1.5);



elseif cur_angle>=des_angle&&abs(diff)<=4.99999&&abs(diff)>=3
   ERB24_LEFT(1)

elseif cur_angle>=des_angle&&abs(diff)<=2.99999&&abs(diff)>=1
   ERB24_LEFT(.1)

elseif  cur_angle>=des_angle&&abs(diff)<=.999999&&abs(diff)>=.2
   ERB24_LEFT(.02)

elseif  cur_angle>=des_angle&&abs(diff)<=.199999&&abs(diff)>=.15
   ERB24_LEFT(.01)

elseif  cur_angle>=des_angle&&abs(diff)<=.1499999&&abs(diff)>=.1
   ERB24_LEFT(.003)




end




if cur_angle<=des_angle&&abs(diff)>=20

   ERB24_RIGHT(6);
   pause(.1)
elseif cur_angle<=des_angle&&abs(diff)<=19.99999&&abs(diff)>=10
```

```
        ERB24_RIGHT(3);
        pause(.1)
    elseif cur_angle<=des_angle&&abs(diff)<=9.99999&&abs(diff)>=5
        ERB24_RIGHT(1.5);


    elseif cur_angle<=des_angle&&abs(diff)<=4.99999&&abs(diff)>=3
        ERB24_RIGHT(1)

    elseif cur_angle<=des_angle&&abs(diff)<=2.99999&&abs(diff)>=1
        ERB24_RIGHT(.1)

         elseif  cur_angle<=des_angle&&abs(diff)<=.999999&&abs(diff)>=.2
             ERB24_RIGHT(.02)

         elseif  cur_angle<=des_angle&&abs(diff)<=.199999&&abs(diff)>=.15
             ERB24_RIGHT(.01)

        elseif cur_angle<=des_angle&&abs(diff)<=.149999&&abs(diff)>=.1
            ERB24_RIGHT(.003)

end

end

-----------------------------------------------------------------------------------------------
function[current_posit] = angle_calibration(analog_posit)

%insert data into x and y to get the calibration curve for rotation.
%the x data used for the angle was taken from digital inclinometer
%the y data was read from the digital pmd1608fs_2


% First calibration prior to connecting all info through Amphenol cord.
%(4/11/05)

%x=[-50.3 -40.3 -29.9 -19.8 -10.2 0 10 20.3 30.1 40.2 50.1];
%y=[1.093 1.350 1.641 1.906 2.160 2.432 2.712 3 3.263 3.530 3.79];

%Second Calibration Data after connecting through Amphenol cord.
%(4/13/05)

x=[-49.8 -40 -29.9 -19.95 -9.8 .2 10.4 20 30.2 40.3 50.4];
x=x+.4;
y=[1.0978 1.3665 1.6335 1.9022 2.1701 2.4381 2.7086 2.9846 3.2541 3.5261 3.8];

%create a square polynomial to fit the data to.
p=polyfit(y,x,2);

%evaluate the polynomial at the analog position and output the position in
%degrees
current_posit=polyval(p,analog_posit);
```

```
-----------------------------------------------------------------------------------------------
function[analog_pressure]=PMD_1608FS_3
%This function is called to take an average of channel one's output.
%It will be used for feedback on the pressure of the three hole probe.

clear all

global analog_pressure



openDAQ=daqfind;
for i=1:length(openDAQ),
    stop(openDAQ(i));
end


%stop(ai);
%delete(ai);
%clear all
ai = analoginput('mcc',0);
%create channel 3-7 (called 2-6)
addchannel(ai,2:6);

set(ai,'SampleRate',500);
ActualAIRate = ai.SampleRate;
set(ai,'SamplesPerTrigger',30)

%the t is left in for plotting and troubleshooting
start(ai)
[a,t] = getdata(ai);
start(ai)
[b,t] = getdata(ai);
start(ai)
[c,t] = getdata(ai);
start(ai)
[d,t] = getdata(ai);
%plot(a,t)

%take an average of channel each output.
analog_pressure=(sum(a)/30+sum(b)/30+sum(C)/30+sum(d)/30)/4;


%stop and delete the ai
stop(ai);
delete(ai);
```

```
-------------------------------------------------------------------------------------------
function[Pressures] = pressure_calibration

%insert data into x and y to get the calibration curve for rotation.
%the x data used for the angle was taken from digital inclinometer
%the y data was read from the digital pmd1608fs_2


% First calibration.
%(5/3/05)
%pressure in inches of mercury converted to Pascals

global Pressures
global analog_pressure

analog_pressure=PMD_1608FS_3;

x=[0 2 4 6 8 10 12 14 16]'*3386.38+[29.795]*3386.38;


%Values for upper pressure connection.

%y(:,1)=[3.50302383
%3.339436942
%3.172269271
%3.010462603
%2.844668243
%2.679341828
%2.519305206
%2.354955367
%2.191195544
%];
%y(:,2)=[3.506919966
%3.34128837
%3.171496147
%3.007471834
%2.83968363
%2.671834389
%2.509325806
%2.342646423
%2.1769131
%];
%y(:,3)=[3.504193688
%3.342549783
%3.176033163
%3.015854124
%2.850883752
%2.686930648
%2.52751456
%2.364995804
%2.202263422
%];
%y(:,4)=[3.512240279
%3.346832481
```

```
%3.17700974
%3.013300781
%2.845888965
%2.678171969
%2.515765113
%2.349106076
%2.183474479
%];

%Values for Upper pressure connection 5_5_05

y(:,1)=[3.5036
3.3389
3.1709
3.0046
2.8464
2.6802
2.5208
2.3544
2.1891
];
y(:,2)=[3.5082
3.341
3.1703
3.0016
2.8412
2.6727
2.5107
2.3419
2.1742
];
y(:,3)=[3.5048
3.3418
3.1744
3.0098
2.8527
2.688
2.5292
2.3641
2.2001
];
y(:,4)=[3.5
3.3348
3.1685
3.0069
2.8396
2.6759
2.5129
2.3494
2.1863
];

y(:,5)=[3.5129
3.3457
3.1751
```

```
3.0065
2.8469
2.6785
2.5167
2.3473
2.1803
];



for ii=1:5

%create a polynomial to fit the data to.  Data was nearly linear.
p(ii,:)=polyfit(y(:,ii),x,1);

%evaluate the polynomial at the analog position and output the position in
%degrees

Pressures(ii)=polyval(p(ii,:),analog_pressure(ii));

end




---------------------------------------------------------------------------------------------------

function[]=OUTPUT_TO_EXCEL(Tp, Tj)
%this function gets the data from the devices using PMD_1608FS_2 and
%PMD_1608FS_3 and sends the data to ExcelWrite to be written to an Excel
%Spreadsheet.




global Pressures




%Freddy.txt is a notepad value used to hold on to data outside of the
%program to ensure headers will be created.
load freddy.txt

%restart.txt is a notepad value used to hold on to data outside of the
%program to ensure data will not be overwritten if the run is restarted.
load restart.txt

%if it is the first run, or the run has been restarted, this loop will
%place the header in and then write the first line of data.
if freddy==1;

   %if first line, put the header in. if restart, add the number of
   %lines so that data will not be overwritten.
```

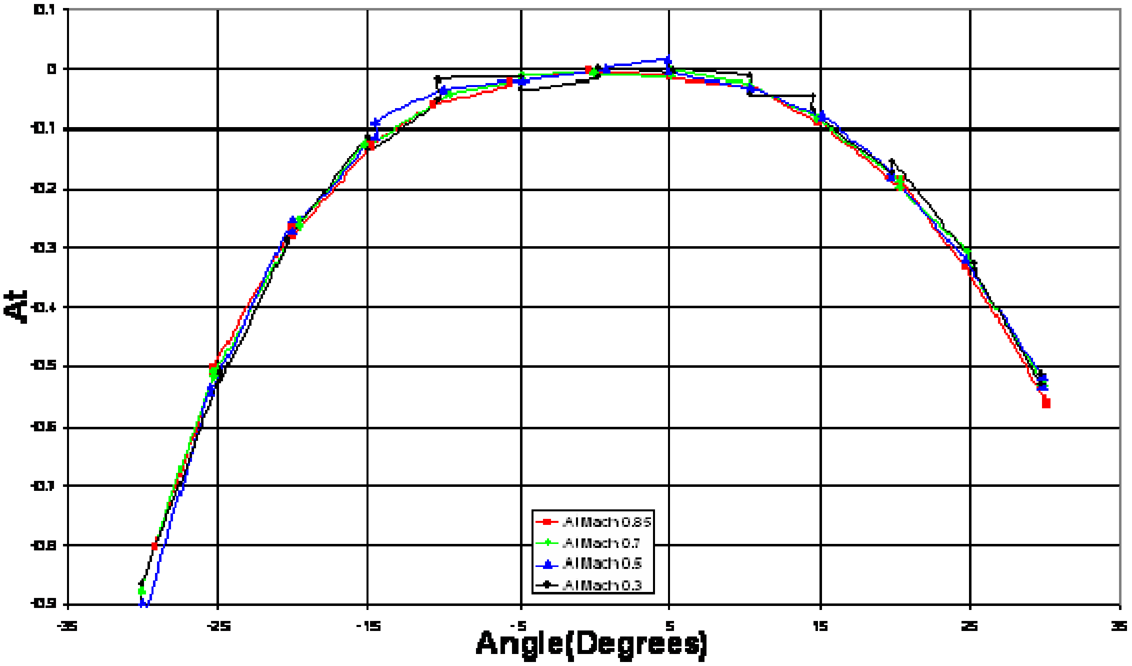
```
           a=freddy+restart;

           %header
           m={'angle', 'P1', 'P2', 'P3', 'Poj', 'Psj', 'Tp', 'Tj', 'q_probe', 'V_jet', 'Cp1_Probe',...
                'Cp2_Probe', 'Cp3_Probe', 'At_Probe', 'As_Probe', 'Mach_No'};



           %create a variable that can be evaluated so that the values of lines
           %can be changed.
           fred = ['ExcelWrite(' "" 'C:\Documents and Settings\user\Desktop\ERB24\testing1.xls' ""
', 0,'...

                "" '[' num2str(a) ',' num2str(2) ']' "" ', m)'];

           %evaluate the variable and send the info to ExcelWrite to evaluate it and
           %send it to an EXCEL Spreadsheet.
        eval(fred);



        %change the row number it will be the next available.



        %Call the calculation function
        calculate(Tp, Tj);

        %create a variable that can be evaluated so that the values of lines
        %can be changed.
        %fred = ['ExcelWrite(' "" 'C:\Documents and Settings\user\Desktop\ERB24\testing1.xls' ""
', 0,'...

        %       "" '[' num2str(a) ',' num2str(2) ']' "" ', m)'];

        %evaluate the variable and send the info to ExcelWrite to evaluate it and
        %send it to an EXCEL Spreadsheet.
        %eval(fred);

        %Change the value of freddy to the next row number.
        %freddy=freddy+1;

        %Save the row number to a text file so that it will not be cleared.
        %save freddy.txt freddy -ascii


        %If the run is not the first, or a reset, just output dats.
        elseif freddy ~=1

           a=freddy+restart;

           %Call the calculation function
           calculate(Tp, Tj);
```

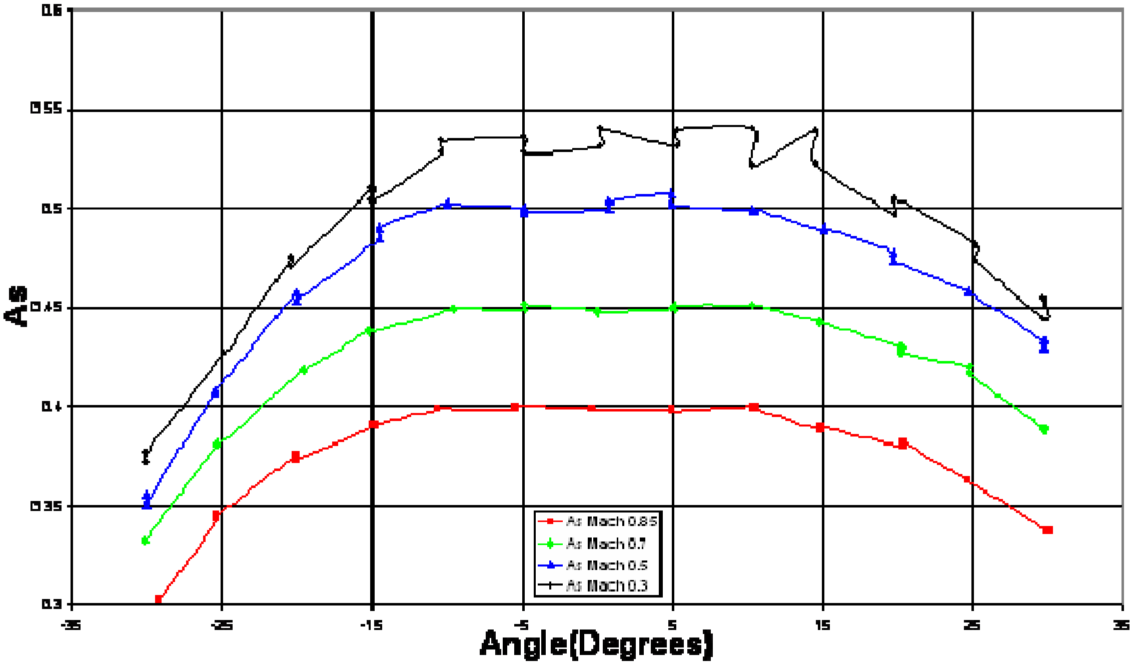
```
            %create a variable that can be evaluated so that the values of lines
            %can be changed.
            %fred = ['ExcelWrite(' '"' 'C:\Documents and Settings\user\Desktop\ERB24\testing1.xls'
'"' ', 0,'...
            %       '"' '[' num2str(a) ',' num2str(2) ']' '"' ', m)'];

            %evaluate the variable and send the info to ExcelWrite to evaluate it and
            %send it to an EXCEL Spreadsheet.
            %eval(fred);

            %Change the value of freddy to next row number.
            %freddy=freddy+1;

            %Save the row number to a text file so that it will not be cleared.
            %save freddy.txt freddy -ascii
        end

        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%
        %SUBFUNCTIONS
        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%

        function[m] = calculate(Tp, Tj)
        %this subfunction will get all required data three times and average
        %the three sets of values before outputting the final answer.

        %%%%%%%%%%%%CONSTANTS%%%%%%%%%%%%%%%%%%
        %Cp of air
        Cp_air=1005;

        %gamma of air
        gam_air=1.41;  %gamma of air

        %gravitational constant (m/sec^2)
        g=9.81; %gravitational constant (m/sec^2)

        %density of Mercury (kg/m^3) (calibration pressure in inches of mercury)
        rho_Hg=13550;



        %loop for data
        for i=1:3

            %Get the data from the PMD_1608FS

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%
            %Get the current angle.
            h=PMD_1608FS_2;
```

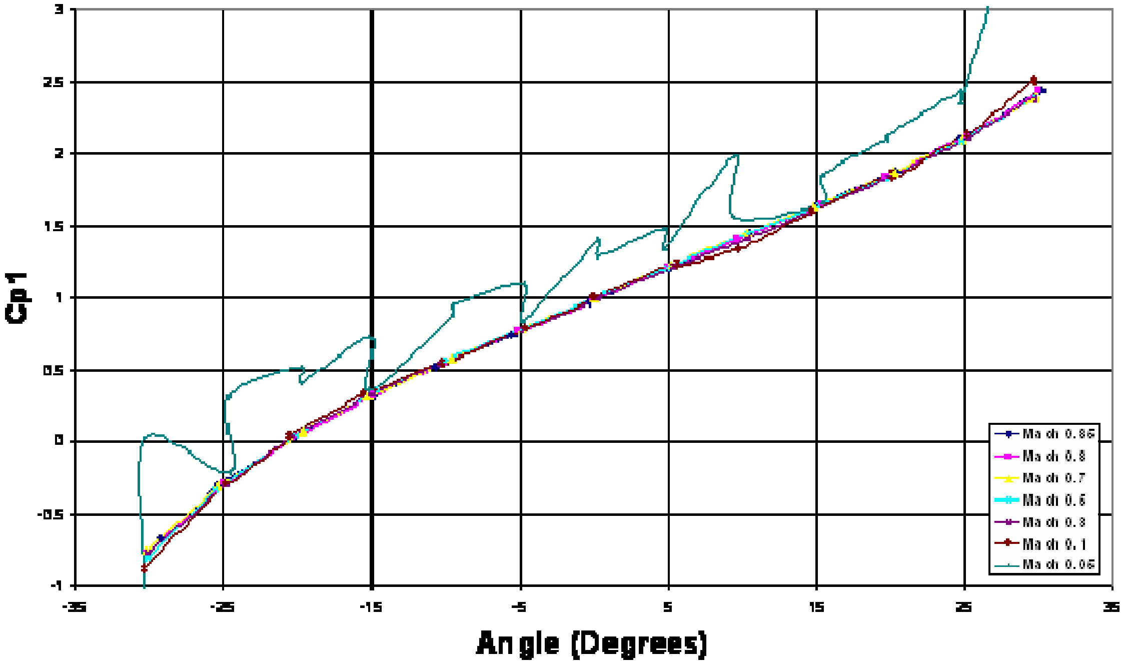
```
          %Convert the angle to degrees.
          cur_angle(i)=angle_calibration(h);

          %get the pressure variable info.

          Pressures=pressure_calibration;
          %Pressures=PMD_1608FS_3;


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%
          %Cobra Probe pressures

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%
          %Left Hole Pressure
          P2=Pressures(1);

          %Center Hole Pressure
          P1=Pressures(2);

          %Right Hole Pressure
          P3=Pressures(3);

          %Total Pressure of jet
          Poj=Pressures(5);

          %Static Pressure of jet
          Psj=Pressures(4);

          %%%%%%%%%%%%%%%%%%%%%%
          %Temperature in pipe
          %Tp=is entered manually in the GUI;
          %%%%%%%%%%%%%%%%%%%%%%%%%

          %%%%%%%%%%%%CALCULATIONS%%%%%%%%%%%%%%%%%%%%


          q_probe=P1-(P2+P3)/2;

          V_jet(i)=sqrt(1-(Psj/Poj)^((gam_air-1)/gam_air))*sqrt(2*Cp_air*Tp);


          Cp1_Probe(i)=(P1-P2)/q_probe;

          Cp2_Probe(i)=(P1-P3)/q_probe;

          Cp3_Probe(i)=(P2-P3)/q_probe;

          At_Probe(i)=(P1-Poj)/q_probe;
          As_Probe(i)=q_probe/(Poj-Psj);

          %Mach Number
          Mach_No(i)=sqrt(((Poj/Psj)^((gam_air-1)/gam_air)-1)*(2/(gam_air-1)));

        end

        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
        %Average the values
        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

        cur_angle=(cur_angle(1)+cur_angle(2)+cur_angle(3))/3;

        Cp1_Probe=(Cp1_Probe(1)+Cp1_Probe(2)+Cp1_Probe(3))/3;

        Cp2_Probe=(Cp2_Probe(1)+Cp2_Probe(2)+Cp2_Probe(3))/3;

        Cp3_Probe=(Cp3_Probe(1)+Cp3_Probe(2)+Cp3_Probe(3))/3;

        V_jet=(V_jet(1)+V_jet(2)+V_jet(3))/3;

        At_Probe= (At_Probe(1)+At_Probe(2)+At_Probe(3))/3;

        As_Probe=(As_Probe(1)+As_Probe(2)+As_Probe(3))/3;

        Mach_No=(Mach_No(1)+Mach_No(2)+Mach_No(3))/3;

        Tj=Tj;

        %Create the data array to be written.
        m=[cur_angle, Pressures(3),Pressures(2),Pressures(1), Pressures(5), Pressures(4), Tp,
Tj, q_probe, V_jet, Cp1_Probe,...
             Cp2_Probe, Cp3_Probe, At_Probe, As_Probe, Mach_No];
        %Freddy.txt is a notepad value used to hold on to data outside of the
        %program to ensure headers will be created.
        load freddy.txt

        %restart.txt is a notepad value used to hold on to data outside of the
        %program to ensure data will not be overwritten if the run is restarted.
        load restart.txt

        a=freddy+1+restart;

          %create a variable that can be evaluated so that the values of lines
          %can be changed.

          fred = ['ExcelWrite(' '''' 'C:\Documents and Settings\user\Desktop\ERB24\testing1.xls' ''''
', 0,'...
                '''' '[' num2str(a) ',' num2str(2) ']' '''' ', m)'];

          %evaluate the variable and send the info to ExcelWrite to evaluate it and
          %send it to an EXCEL Spreadsheet.
```

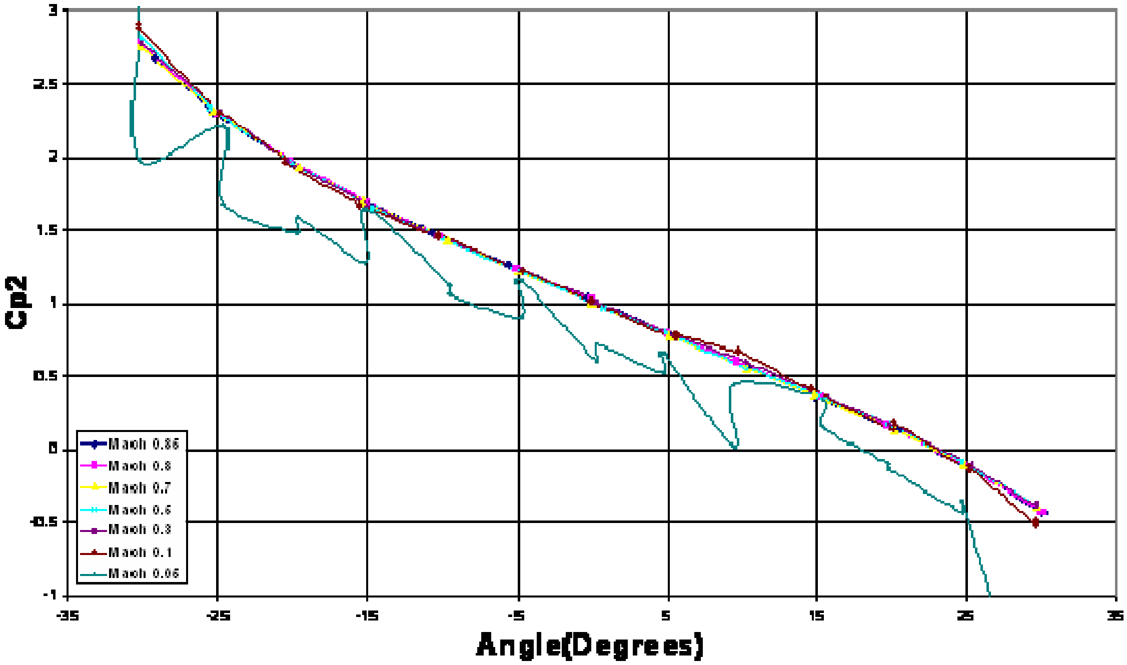
```
eval(fred);

%Change the value of freddy to next row number.
freddy=freddy+1;

%Save the row number to a text file so that it will not be cleared.
save freddy.txt freddy -ascii
```

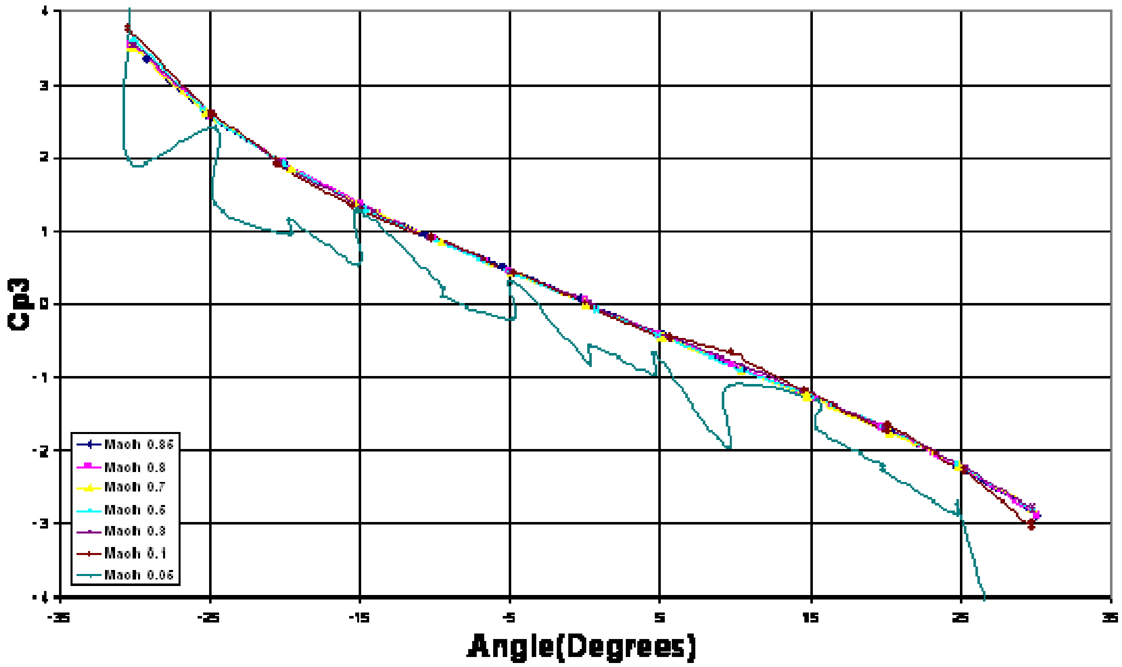
APPENDIX G: SCANIVALVE CALIBRATION GRAPHS





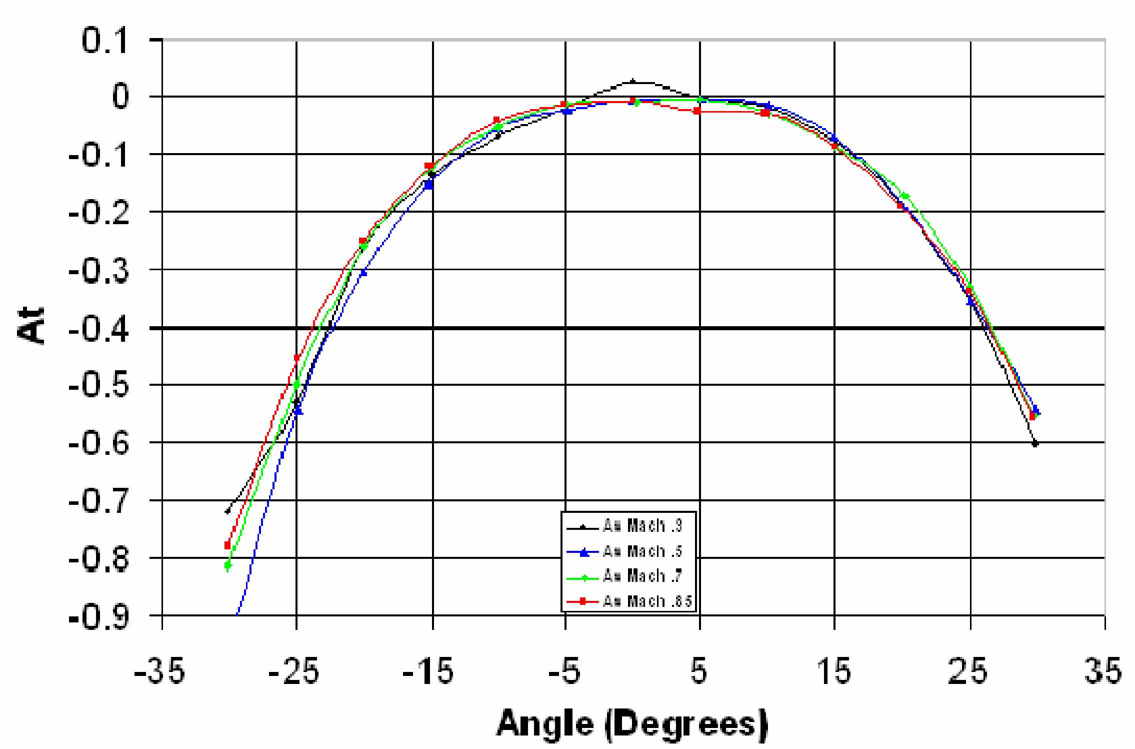

Cp1
3
2.5
2
1.5
1
0.5
0
-0.5
-1
-35
-25
-15
-5
5
15
25
35
Angle (Degrees)
Mach 0.86
Mach 0.8
Mach 0.7
Mach 0.6
Mach 0.3
Mach 0.1
Mach 0.06

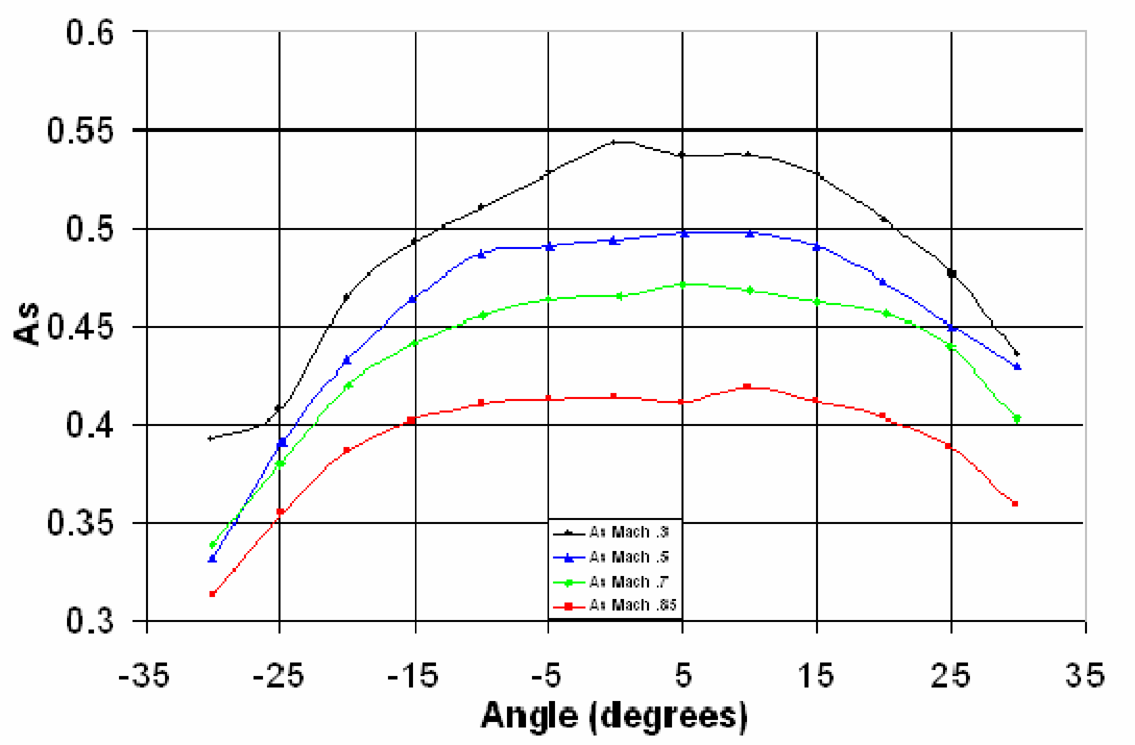

Cp2
3
2.5
2
1.5
1
0.5
0
-0.5
-1
-35
-25
-15
-5
5
15
25
35
Angle(Degrees)
Mach 0.86
Mach 0.8
Mach 0.7
Mach 0.6
Mach 0.3
Mach 0.1
Mach 0.06

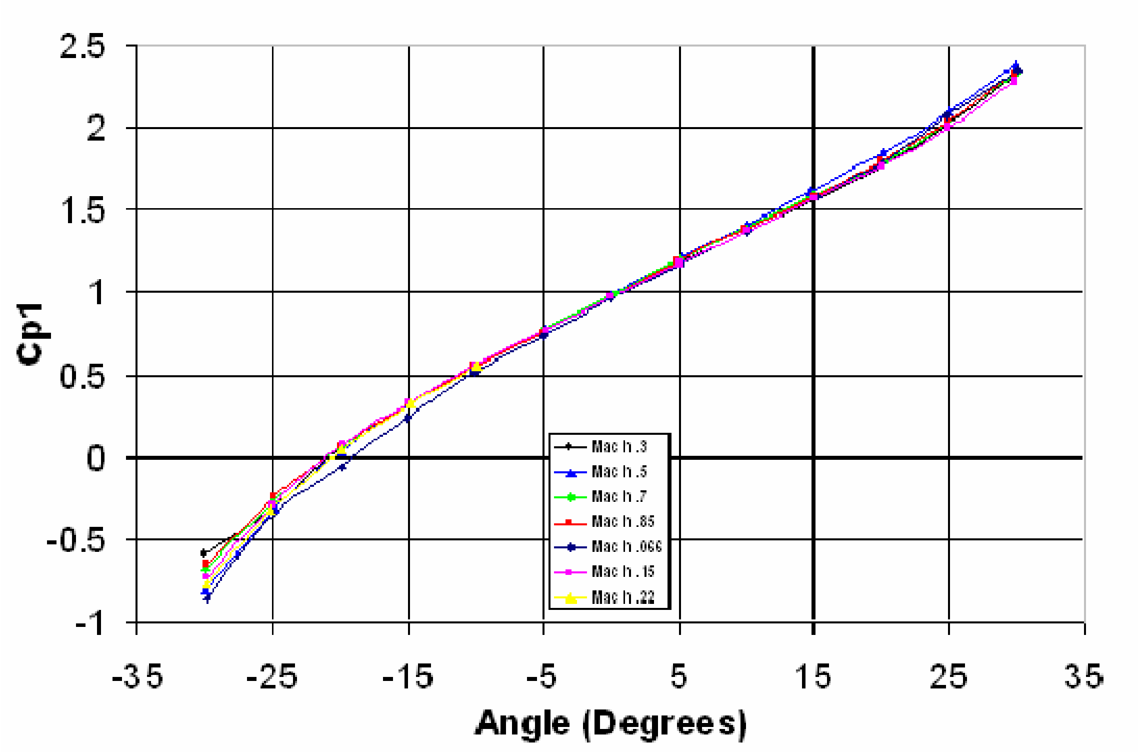

Cp3
4
3
2
1
0
-1
-2
-3
-4
-35
-25
-15
-5
5
15
25
35
Angle(Degrees)
Mach 0.86
Mach 0.8
Mach 0.7
Mach 0.6
Mach 0.3
Mach 0.1
Mach 0.06

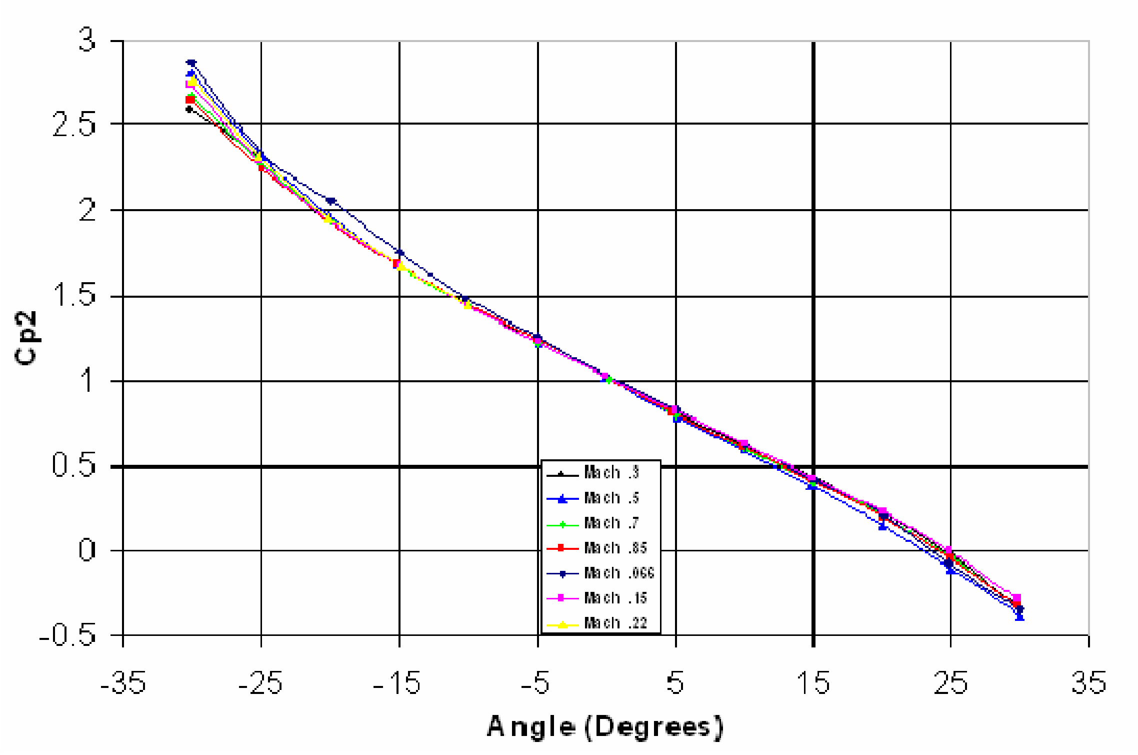
THIS PAGE INTENTIONALLY LEFT BLANK

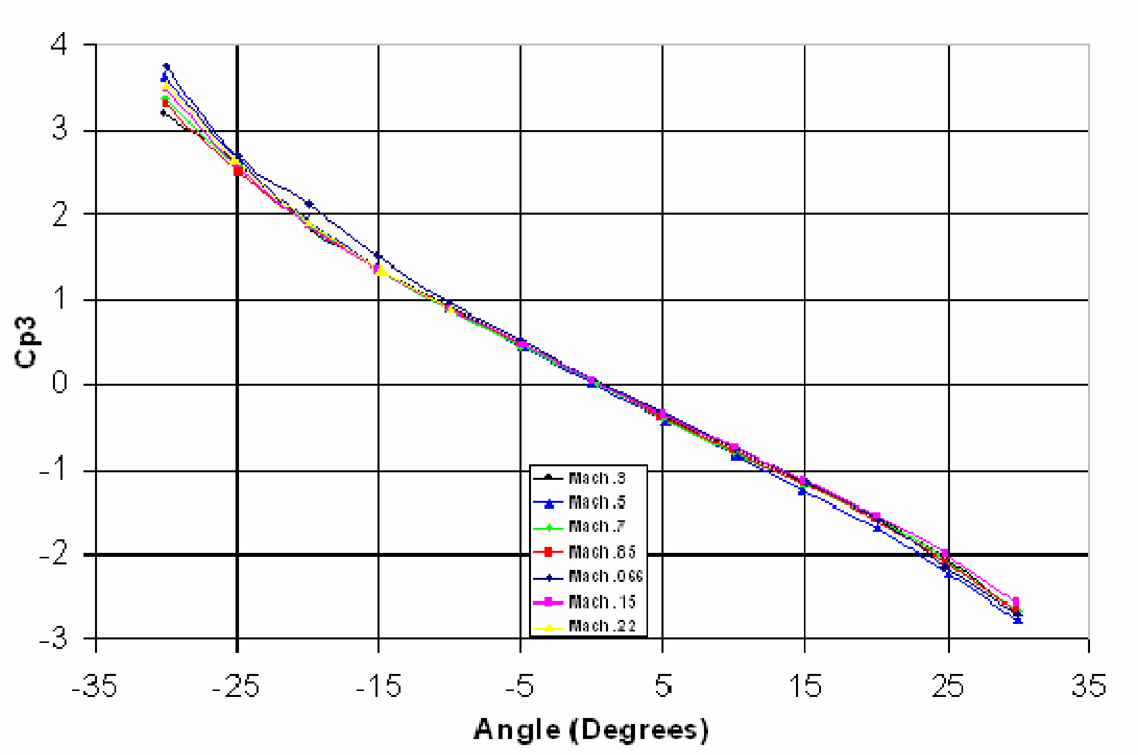
APPENDIX H: PX-138 CALIBRATION GRAPHS






2.5
2
1.5
1
0.5
0
-0.5
-1
Cp1
-35
-25
-15
-5
5
15
25
35
Angle (Degrees)
Mach .3
Mach .5
Mach .7
Mach .85
Mach .066
Mach .15
Mach .22


3
2.5
2
1.5
1
0.5
0
-0.5
Cp2
-35
-25
-15
-5
5
15
25
35
Angle (Degrees)
Mach .3
Mach .5
Mach .7
Mach .85
Mach .066
Mach .15
Mach .22


4
3
2
1
0
-1
-2
-3
Cp3
Mach .3
Mach .5
Mach .7
Mach .85
Mach .066
Mach .15
Mach .22
-35
-25
-15
-5
5
15
25
35
Angle (Degrees)

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX I: MATLAB INTERPOLATION FILES

```
% m-file to call the 3 hole probe output file

clear all

% Type of file to get data from
%file_type = 'e';  % Call from excel
file_type = 'm';   % Call from text file

% Measured values, can be changed to a function later on
p1_m  = [93692.44746
93648.41833
93540.03893
93419.38217
93308.88599
93144.62346
93158.17088
93047.25134
93104.40454
93128.95925
93103.98118
93125.99575
92763.60213
92046.01196
91071.86744
];
p2_m  = [95925.86792
95915.51443
95922.96894
95927.52448
95932.08001
95862.50458
95918.82755
95897.70643
95905.98922
95862.91872
95831.44412
95680.69734
94890.93334
93210.3553
91121.43572
];
p3_m  = [93731.43686
93648.22584
93532.23472
93429.69185
93297.73094
93106.09345
93086.76159
93009.85444
93100.6301
93164.089
93188.46395
```

```
      93155.26359
      92857.30091
      92044.94283
      91100.62385
      ];
      poj_m = [95852.60365
      95850.08642
      95877.35637
      95901.27002
      95863.51163
      95820.71878
      95880.71268
      95850.9255
      95881.97129
      95873.58053
      95845.47151
      95879.03452
      95921.40784
      95846.73012
      95884.06898
      ];

      %p1_m  = 102794.6236;
      %p2_m  = 107286.9565;
      %p3_m  = 105166.2278;
      %poj_m = 107363.8443;

      tic
      %[Pitch_m,Yaw_m,Mach_m,ps_m,po_m]                                =
three_hole_probe_output(file_type,p1_m,p2_m,p3_m,poj_m)

      [Pitch_m_2,Yaw_m_2,Mach_m_2,ps_m_2,po_m_2]                       =
three_hole_probe_output(file_type,p1_m,p2_m,p3_m,p2_m)

      toc


      -------------------------------------------------------------------

      % M-function file to obtain the flow direction and mach number
based on the input data from a 3-hole probe
      %
      % To assume ZERO PITCH, just pass p2_m to poj_m so that they are
the same. This will stop the program
      % searching along this dimension.
      %
      % To load data from the excel spreadsheets they must be moved
into the same directory, the constants file
      % will then also be updated
      % file_type = 'e' (excek) or 'm' (matlab)


      function            [Pitch_m,Yaw_m,Mach_m,ps_m,po_m]            =
three_hole_probe_output(file_type,p1_m,p2_m,p3_m,poj_m)
```

```
      % If only one input point is needed two points are required to
make griddata3 work
      if length(p1_m)<2
          point_no = 1;
          p1_m   = [p1_m   p1_m]; p2_m   = [p2_m   p2_m]; p3_m = [p3_m
p3_m]; poj_m = [poj_m poj_m];
      else
          point_no = 0;
      end

      % Input matricies or vectors are reshaped to column vectors
      shape_p1_m           =      size(p1_m);           p1_m           =
reshape(p1_m,1,shape_p1_m(1)*shape_p1_m(2));
      shape_p2_m           =      size(p2_m);           p2_m           =
reshape(p2_m,1,shape_p2_m(1)*shape_p2_m(2));
      shape_p3_m           =      size(p3_m);           p3_m           =
reshape(p3_m,1,shape_p3_m(1)*shape_p3_m(2));
      shape_poj_m         =         size(poj_m);         poj_m         =
reshape(poj_m,1,shape_poj_m(1)*shape_poj_m(2));

      % Physical constants
      Gam_gas = 1.41;

      %Measured constants
      q_yaw_m   = p1_m-((p2_m+p3_m)/2);
      q_pitch_m = poj_m-((p2_m+p3_m+p1_m)/3);

      Cp1_m = (p1_m-p2_m)./q_yaw_m;
      Cp2_m = (p1_m-p3_m)./q_yaw_m;
      Cp3_m = (p3_m-p2_m)./q_yaw_m;     % Side holes
      Cp4_m = (poj_m-p2_m)./q_pitch_m;
      Cp5_m = (poj_m-p3_m)./q_pitch_m;
      Cp6_m = (poj_m-p1_m)./q_pitch_m;  % Centre hole

      %-------------------------------------------------------------
----------------------------------------------
      if file_type == 'e'
          % Coefficients of pressure are read in but are calculated
later from the raw data.
          %Cp1 = Load_constant(12)
          %Cp2 = Load_constant(13)
          %Cp3 = Load_constant(14)

          Yaw = Load_constant(2);

          % Pressure are read in
          p1  = Load_constant(4);
          p2  = Load_constant(3);
          p3  = Load_constant(5);
          poj = Load_constant(6);

          % At, As and Mach number are loaded in
          At   = Load_constant(15);
          As   = Load_constant(16);
          Mach = Load_constant(17);
          % Pitches are constructed
          Pitch = Load_constant(18);

          save 3_hole_constants Yaw p1 p2 p3 poj At As Mach Pitch
      else
          load 3_hole_constants Yaw p1 p2 p3 poj At As Mach Pitch;
      end % if file_type == 'excel'

      % q coefficients
      q_yaw    = p1-((p2+p3)/2);       % Same one as is used in the
calibration
      q_pitch = poj-((p2+p3+p1)/3);

      %At = (p1 - poj)./q_yaw  % There is some difference between the
spreadsheet and these values

      % Pitch presssure coefficients are now defined for a '4' hole
probe
      Cp1 = (p1-p2)./q_yaw;
      Cp2 = (p1-p3)./q_yaw;
      Cp3 = (p3-p2)./q_yaw;     % Side holes
      Cp4 = (poj-p2)./q_pitch;
      Cp5 = (poj-p3)./q_pitch;
      Cp6 = (poj-p1)./q_pitch;  % Centre hole

      %-----------------------------------------------------------------
---------------------------------------------
      % Initial estimate of Mach number
      ps_m    = min([p1_m; p2_m; p3_m]); %ps_m    = min([p1_m; p2_m;
p3_m; poj_m]);
      po_m    = max([p1_m; p2_m; p3_m]); %po_m    = max([p1_m; p2_m;
p3_m; poj_m]);
      Mach_m        =        sqrt((2/(Gam_gas-1))*((ps_m./po_m).^((1-
Gam_gas)/Gam_gas)-1));

      Pitch_m = 0*ones(size(p1_m)); % Initial estimate of pitch in
degrees
      Yaw_m   = 0*ones(size(p1_m)); % Initial estimates of pitch


      for i = 1:length(Mach_m)
          if (max(max(Mach)) < Mach_m(i))      | (min(min(Mach)) >
Mach_m(i)) |...
                  (max(max(Cp3)) < Cp3_m(i)) | (min(min(Cp3)) >
Cp3_m(i))  |...
                  (max(max(Cp6)) < Cp6_m(i)) | (min(min(Cp6)) >
Cp6_m(i))

              disp('Warning, data may be outside calibration range')
              Yaw_m(i) = 0/0;

          end % if (max(max(Mach)) < max(Mach_m))
      end
      temp = find((isnan(Yaw_m))~=1); % Data that is within calibration
range

      % Initial estimate of yaw
      Yaw_m(temp)                                                      =
griddata3(Mach,Pitch,Cp3,Yaw,Mach_m(temp),Pitch_m(temp),Cp3_m(temp));

      % Variables on the tolerance
      tol     =  0.25;  Pitch_m_old  =  Pitch_m+2*tol;  Yaw_m_old  =
Yaw_m+2*tol;
      count = 0;

      while      (max([max(abs(Yaw_m-Yaw_m_old))      max(abs(Pitch_m-
Pitch_m_old))]) > tol) & count < 10
          % Stored variables for the tolerance
          Pitch_m_old = Pitch_m;  Yaw_m_old = Yaw_m;
          count = count+1;

          if length(intersect(poj_m,p2_m)) == 0 % is this is true then
it is assumed that the pitch is zero
              % Pitch is calculated using the centre hole but in case
this is outside the range the other two are used
              Pitch_m(temp)                                            =
griddata3(Yaw,Cp6,Mach,Pitch,Yaw_m(temp),Cp6_m(temp),Mach_m(temp))

              if isnan(Pitch_m(temp))
                  Pitch_m(temp)                                        =
griddata3(Yaw,Cp5,Mach,Pitch,Yaw_m(temp),Cp5_m(temp),Mach_m(temp));
              end
              if isnan(Pitch_m(temp))
                  Pitch_m(temp)                                        =
griddata3(Yaw,Cp4,Mach,Pitch,Yaw_m(temp),Cp4_m(temp),Mach_m(temp));
              end
          end

          % Yaw is corrected to the right pitch
          Yaw_m(temp)                                                  =
griddata3(Mach,Pitch,Cp3,Yaw,Mach_m(temp),Pitch_m(temp),Cp3_m(temp));

          % The static and stagnation pressure constants are found
          At_m(temp)                                                   =
griddata3(Mach,Pitch,Yaw,At,Mach_m(temp),Pitch_m(temp),Yaw_m(temp));
          As_m(temp)                                                   =
griddata3(Mach,Pitch,Yaw,As,Mach_m(temp),Pitch_m(temp),Yaw_m(temp));

          %  The  total  pressure  from  the  constants  is  calculated  even
though for the 3-hole probe it is directly measured
          po_m(temp) = p2_m(temp) - At_m(temp).*q_yaw_m(temp);

          % The static pressure is calculated
          ps_m(temp) = po_m(temp) - q_yaw_m(temp)./As_m(temp);

          % The mach number is updated
          Mach_m(temp)                =                sqrt((2/(Gam_gas-
1))*((ps_m(temp)./po_m(temp)).^((1-Gam_gas)/Gam_gas)-1));

      end % while abs(Pitch_m-Pitch_m_old)>tol

      % Points are reshaped to the original input matrix
      if point_no == 1
          Pitch_m = Pitch_m(1);
          Yaw_m   = Yaw_m(1);
          Mach_m  = Mach_m(1);
          ps_m    = ps_m(1);
          po_m    = po_m(1);
      else
          Pitch_m = reshape(Pitch_m,shape_p1_m);
          Yaw_m   = reshape(Yaw_m,shape_p1_m);
          Mach_m  = reshape(Mach_m,shape_p1_m);
          ps_m    = reshape(ps_m,shape_p1_m);
          po_m    = reshape(po_m,shape_p1_m);
      end % if point_no == 1

      %----------------------------------------------------------------
----------------------------------------------
      if 0
          % Cp1 and Cp2 are plotted
          figure(1); close; figure(1);
          plot(Yaw,Cp1,Yaw,Cp2)
          temp = axis; axis([-30 30 temp(3) temp(4)]);

          % Cp3 is plotted
          figure(2); close; figure(2);
          plot(Yaw,Cp3)
          temp = axis; axis([-30 30 temp(3) temp(4)]);

          % Cp4 and Cp6 is plotted
          figure(3); close; figure(3);
          plot(Pitch',Cp4',Pitch',Cp5')
          temp = axis; axis([-30 30 temp(3) temp(4)]);

          % Cp6 is plotted
          figure(4); close; figure(4);
          plot(Pitch',Cp6')
          temp = axis; axis([-30 30 temp(3) temp(4)]);

          % Surface plot of the data is plotted
          figure(5); close; figure(5);
          TRI = delaunay(Yaw,Pitch);
          trisurf(TRI,Yaw,Pitch,Cp3)
          hold on
          %plot3(Yaw_m,Pitch_m,Cp3_m,'or')
      end % if 0


      ---------------------------------------------------------------
      % m-function file to pull in certain data sets
      function [A] = Load_constant(column)

      % Mach 0.06 Nominal
      A(:,1)                                                            =
xlsreadcells('5_10_05_40deg_Pitch.xls','Sheet1',[column],[110:122]);
% 40 Degree pitch
      A(:,end+1)                                                        =
xlsreadcells('5_9_05_30deg_Pitch.xls','Sheet1',[column],[1:13]);
% 30 Degree pitch
      A(:,end+1)                                                        =
xlsreadcells('5_9_05_20deg_Pitch.xls','Sheet1',[column],[1:13]);
% 20 Degree pitch
      A(:,end+1)                                                        =
xlsreadcells('5_9_05_10deg_Pitch.xls','Sheet1',[column],[1:13]);
% 10 Degree pitch
      A(:,end+1)                                                        =
xlsreadcells('5_5_05_0deg_Pitch.xls','Sheet1',[column],[111:123]);
% 0 Degree pitch
      A(:,end+1)                                                        =
xlsreadcells('5_10_05_10deg_Pitch_forward.xls','Sheet1',[column],[104:1
16]); % -10 Degree pitch

      % Mach 0.15 Nominal
      A(:,end+1)                                                        =
xlsreadcells('5_10_05_40deg_Pitch.xls','Sheet1',[column],[92:104]);
% 40 Degree pitch
      A(:,end+1)                                                        =
xlsreadcells('5_9_05_30deg_Pitch.xls','Sheet1',[column],[15:27]);
% 30 Degree pitch
      A(:,end+1)                                                        =
xlsreadcells('5_9_05_20deg_Pitch.xls','Sheet1',[column],[15:27]);
% 20 Degree pitch
      A(:,end+1)                                                        =
xlsreadcells('5_9_05_10deg_Pitch.xls','Sheet1',[column],[16:28]);
% 10 Degree pitch
      A(:,end+1)                                                        =
xlsreadcells('5_5_05_0deg_Pitch.xls','Sheet1',[column],[94:106]);
% 0 Degree pitch
      A(:,end+1)                                                        =
xlsreadcells('5_10_05_10deg_Pitch_forward.xls','Sheet1',[column],[87:99
]); % -10 Degree pitch

      % Mach 0.22 Nominal
      A(:,end+1)                                                        =
xlsreadcells('5_10_05_40deg_Pitch.xls','Sheet1',[column],[76:88]);
% 40 Degree pitch
      A(:,end+1)                                                        =
xlsreadcells('5_9_05_30deg_Pitch.xls','Sheet1',[column],[29:41]);
% 30 Degree pitch
      A(:,end+1)                                                        =
xlsreadcells('5_9_05_20deg_Pitch.xls','Sheet1',[column],[29:41]);
% 20 Degree pitch
      A(:,end+1)                                                        =
xlsreadcells('5_9_05_10deg_Pitch.xls','Sheet1',[column],[30:42]);
% 10 Degree pitch
```

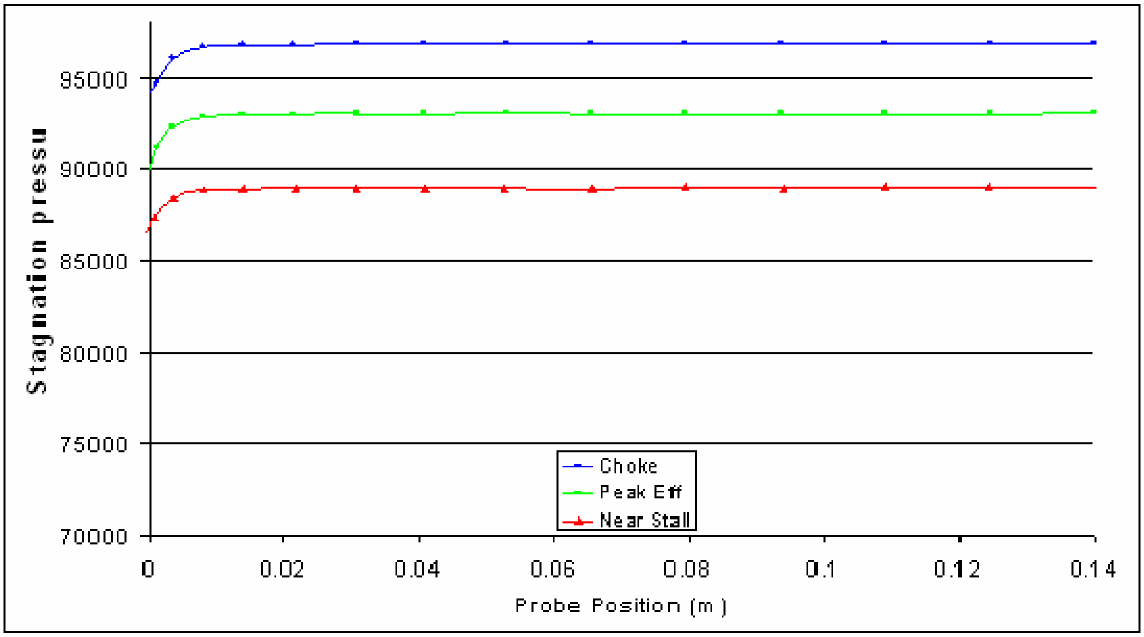
```
      A(:,end+1)                                                       =
xlsreadcells('5_5_05_0deg_Pitch.xls','Sheet1',[column],[77:89]);
% 0 Degree pitch
      A(:,end+1)                                                       =
xlsreadcells('5_10_05_10deg_Pitch_forward.xls','Sheet1',[column],[69:81
]); % -10 Degree pitch

      % Mach 0.3  Nominal
      A(:,end+1)                                                       =
xlsreadcells('5_10_05_40deg_Pitch.xls','Sheet1',[column],[3:15]);
% 40 Degree pitch
      A(:,end+1)                                                       =
xlsreadcells('5_9_05_30deg_Pitch.xls','Sheet1',[column],[43:55]);
% 30 Degree pitch
      A(:,end+1)                                                       =
xlsreadcells('5_9_05_20deg_Pitch.xls','Sheet1',[column],[43:55]);
% 20 Degree pitch
      A(:,end+1)                                                       =
xlsreadcells('5_9_05_10deg_Pitch.xls','Sheet1',[column],[45:57]);
% 10 Degree pitch
      A(:,end+1)                                                       =
xlsreadcells('5_5_05_0deg_Pitch.xls','Sheet1',[column],[1:13]);
% 0 Degree pitch
      A(:,end+1)                                                       =
xlsreadcells('5_10_05_10deg_Pitch_forward.xls','Sheet1',[column],[1:13]
); % -10 Degree pitch

      % Mach 0.5  Nominal
      A(:,end+1)                                                       =
xlsreadcells('5_10_05_40deg_Pitch.xls','Sheet1',[column],[22:34]);
% 40 Degree pitch
      A(:,end+1)                                                       =
xlsreadcells('5_9_05_30deg_Pitch.xls','Sheet1',[column],[60:72]);
% 30 Degree pitch
      A(:,end+1)                                                       =
xlsreadcells('5_9_05_20deg_Pitch.xls','Sheet1',[column],[60:72]);
% 20 Degree pitch
      A(:,end+1)                                                       =
xlsreadcells('5_9_05_10deg_Pitch.xls','Sheet1',[column],[64:76]);
% 10 Degree pitch
      A(:,end+1)                                                       =
xlsreadcells('5_5_05_0deg_Pitch.xls','Sheet1',[column],[21:33]);
% 0 Degree pitch
      A(:,end+1)                                                       =
xlsreadcells('5_10_05_10deg_Pitch_forward.xls','Sheet1',[column],[17:29
]); % -10 Degree pitch

      % Mach 0.7  Nominal
      A(:,end+1)                                                       =
xlsreadcells('5_10_05_40deg_Pitch.xls','Sheet1',[column],[39:51]);
% 40 Degree pitch
      A(:,end+1)                                                       =
xlsreadcells('5_9_05_30deg_Pitch.xls','Sheet1',[column],[77:89]);
% 30 Degree pitch
```

```
      A(:,end+1)                                                       =
xlsreadcells('5_9_05_20deg_Pitch.xls','Sheet1',[column],[77:89]);
% 20 Degree pitch
      A(:,end+1)                                                       =
xlsreadcells('5_9_05_10deg_Pitch.xls','Sheet1',[column],[82:94]);
% 10 Degree pitch
      A(:,end+1)                                                       =
xlsreadcells('5_5_05_0deg_Pitch.xls','Sheet1',[column],[40:52]);
% 0 Degree pitch
      A(:,end+1)                                                       =
xlsreadcells('5_10_05_10deg_Pitch_forward.xls','Sheet1',[column],[34:46
]); % -10 Degree pitch

      % Mach 0.85 Nominal
      A(:,end+1)                                                       =
xlsreadcells('5_10_05_40deg_Pitch.xls','Sheet1',[column],[59:71]);
% 40 Degree pitch
      A(:,end+1)                                                       =
xlsreadcells('5_9_05_30deg_Pitch.xls','Sheet1',[column],[93:105]);
% 30 Degree pitch
      A(:,end+1)                                                       =
xlsreadcells('5_9_05_20deg_Pitch.xls','Sheet1',[column],[94:106]);
% 20 Degree pitch
      A(:,end+1)                                                       =
xlsreadcells('5_9_05_10deg_Pitch.xls','Sheet1',[column],[101:113]);
% 10 Degree pitch
      A(:,end+1)                                                       =
xlsreadcells('5_5_05_0deg_Pitch.xls','Sheet1',[column],[59:71]);
% 0 Degree pitch
      A(:,end+1)                                                       =
xlsreadcells('5_10_05_10deg_Pitch_forward.xls','Sheet1',[column],[52:64
]); % -10 Degree pitch

      %A = reshape(A,[size(A,1)*size(A,2) 1]);
```

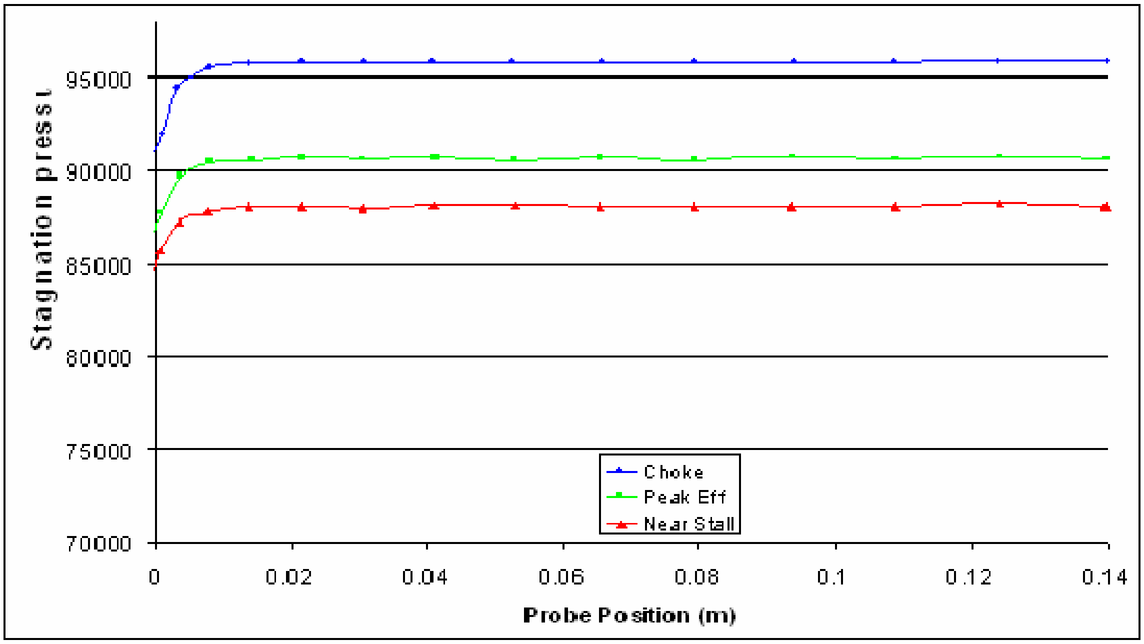
THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX J: UPSTREAM PROBE DATA: STAGNATION PRESSURE

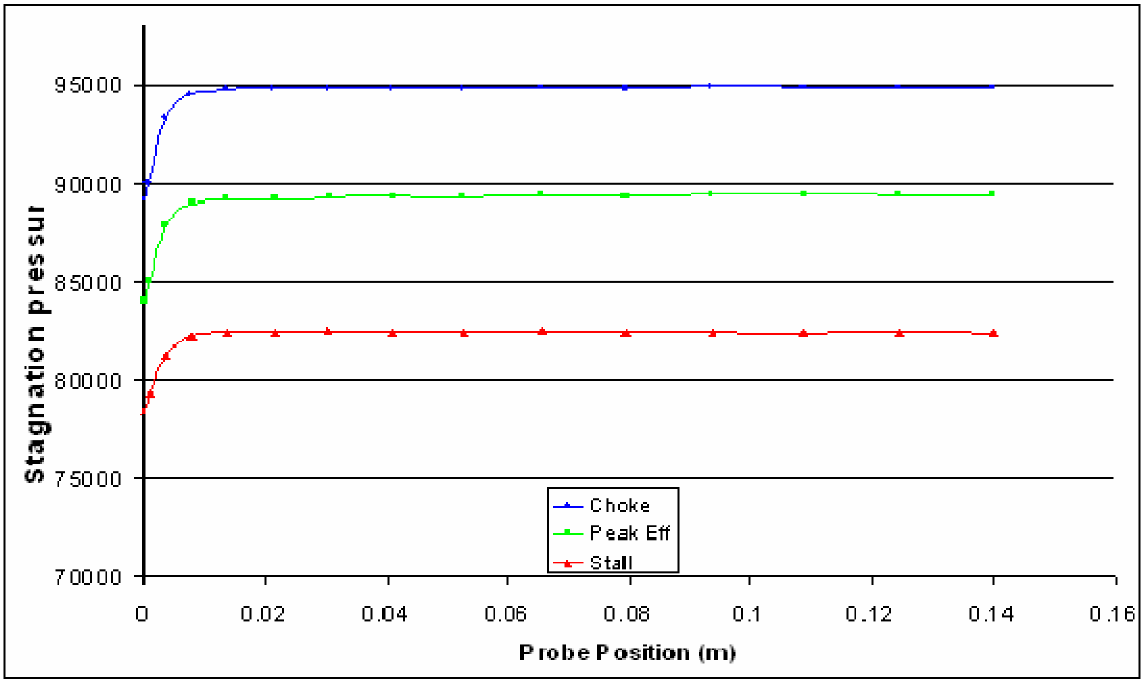
70 % Rotor Speed



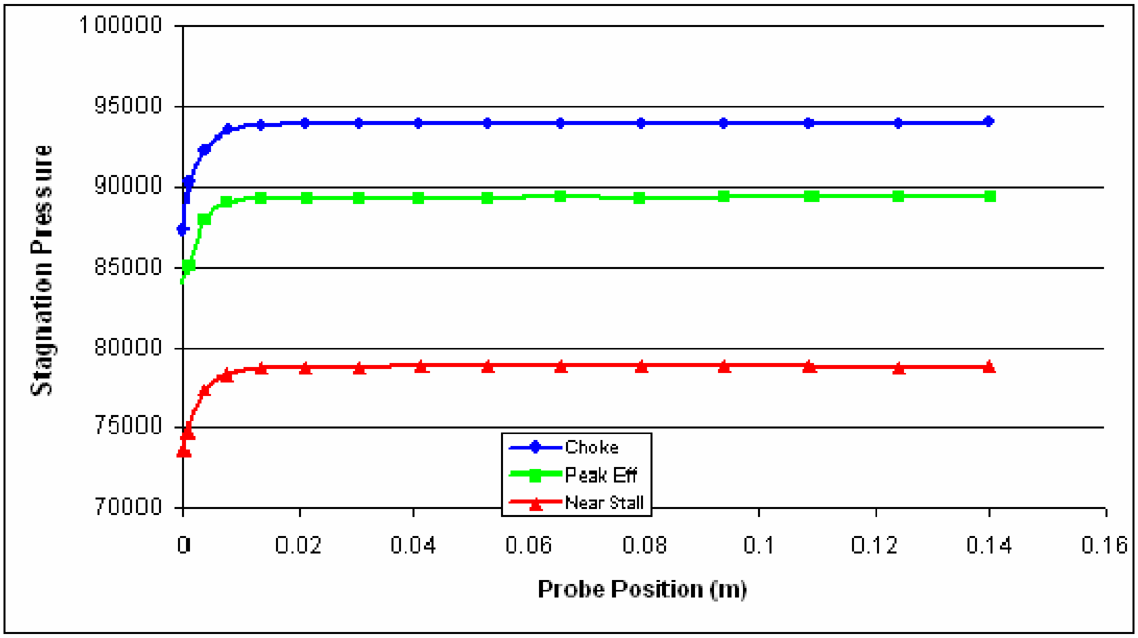
80% Rotor Speed



90% Rotor Speed

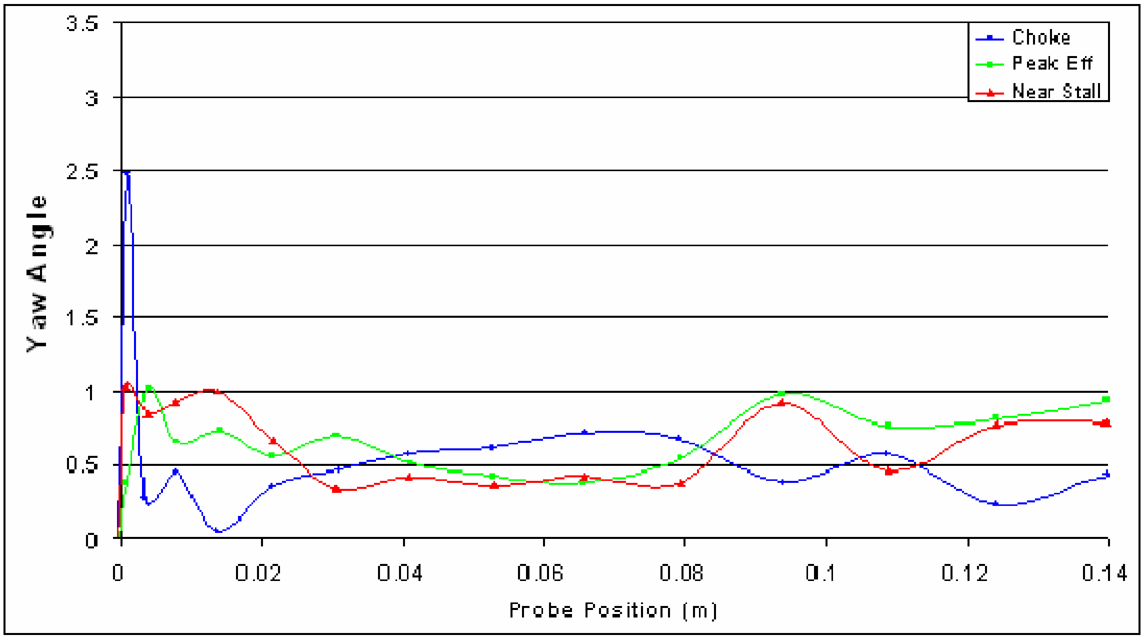


100% Rotor Speed

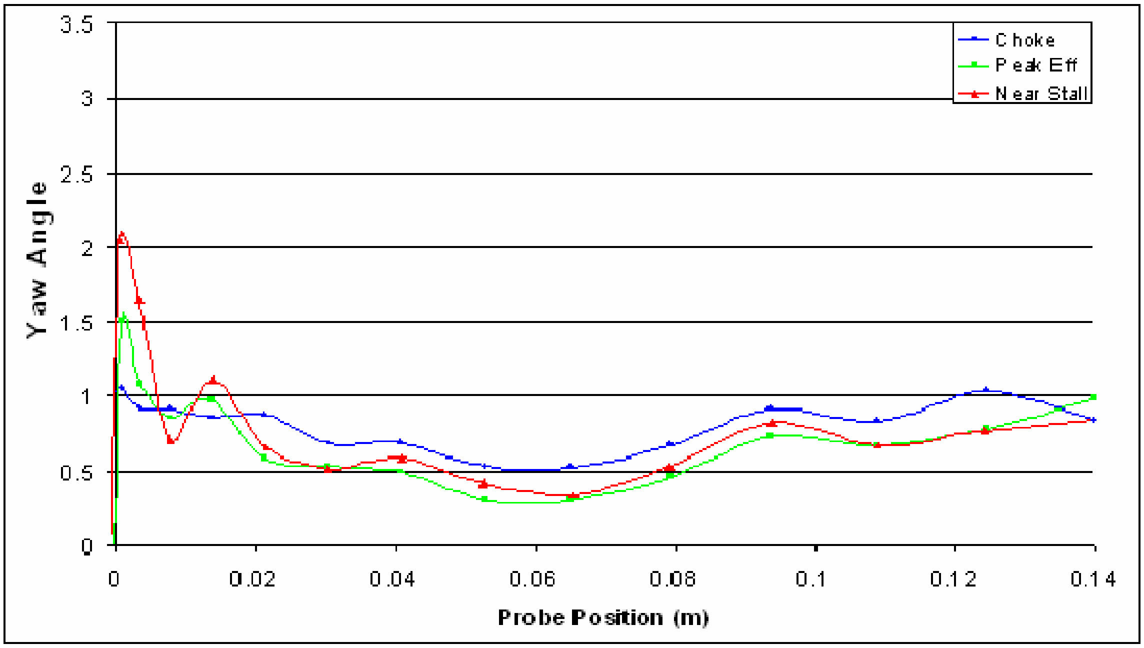


APPENDIX K: UPSTREAM PROBE DATA: YAW ANGLE DISTRIBUTION

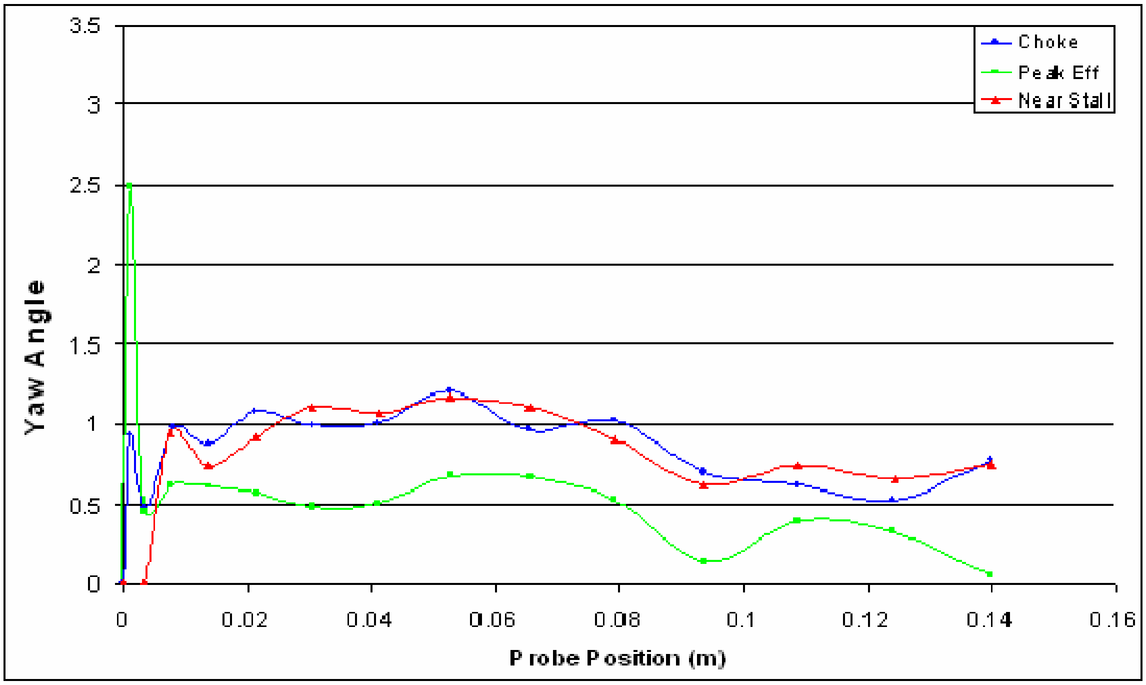
70 % Rotor Speed



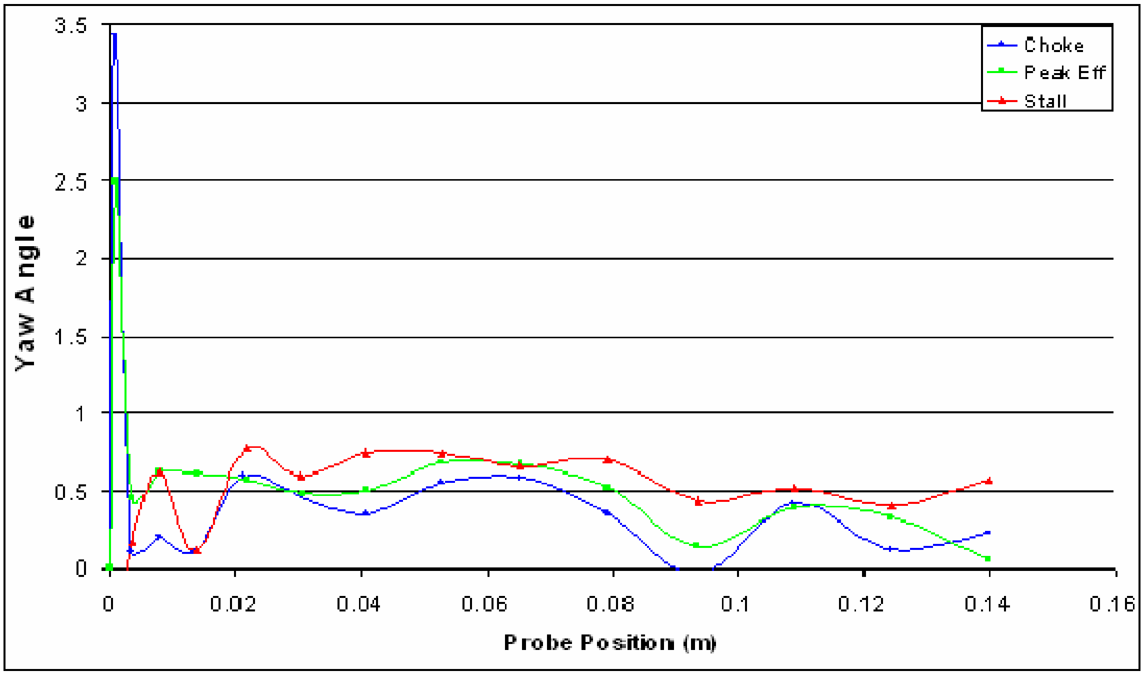
80 % Rotor Speed



90 % Rotor Speed

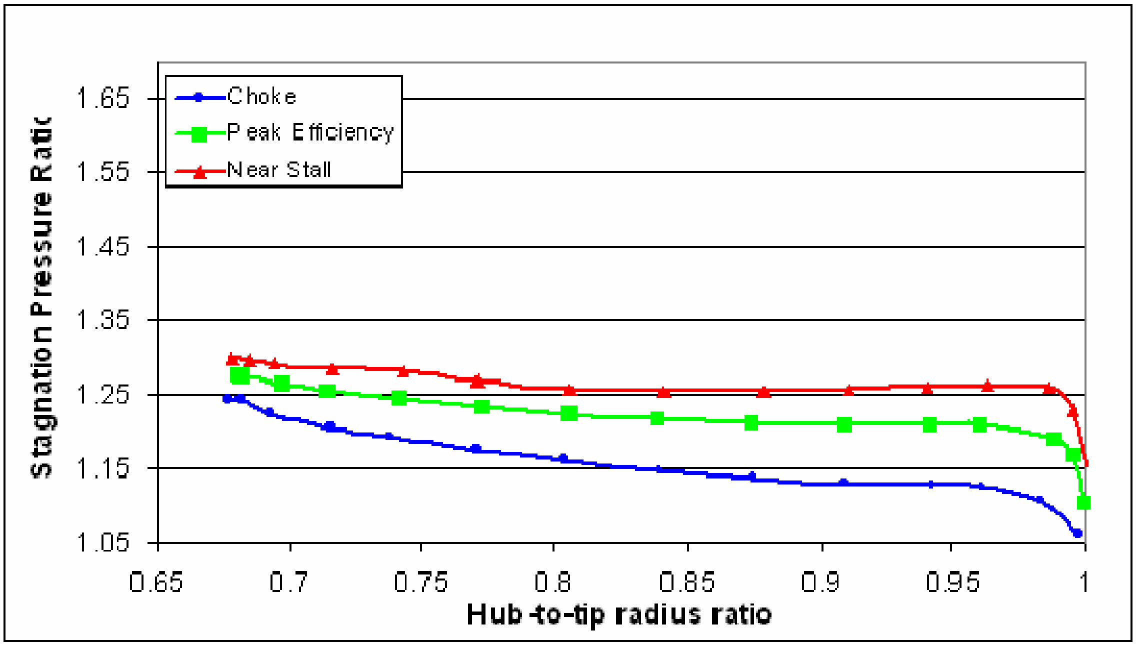


100 % Rotor Speed

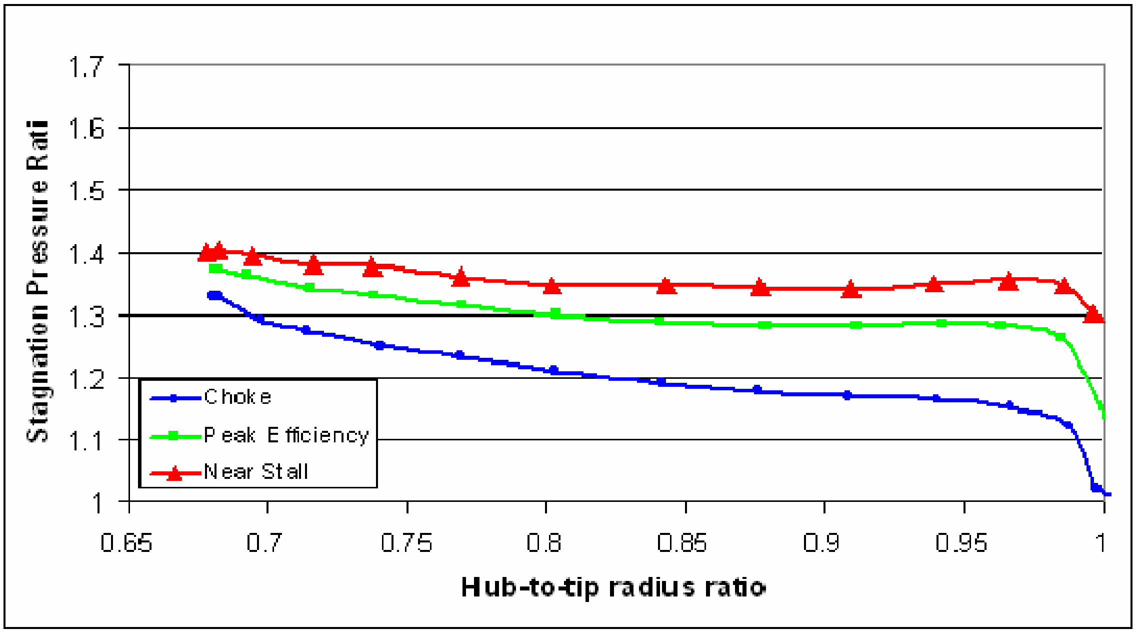


APPENDIX L: DOWNSTREAM PROBE DATA: STAGNATION PRESSURE RATIO

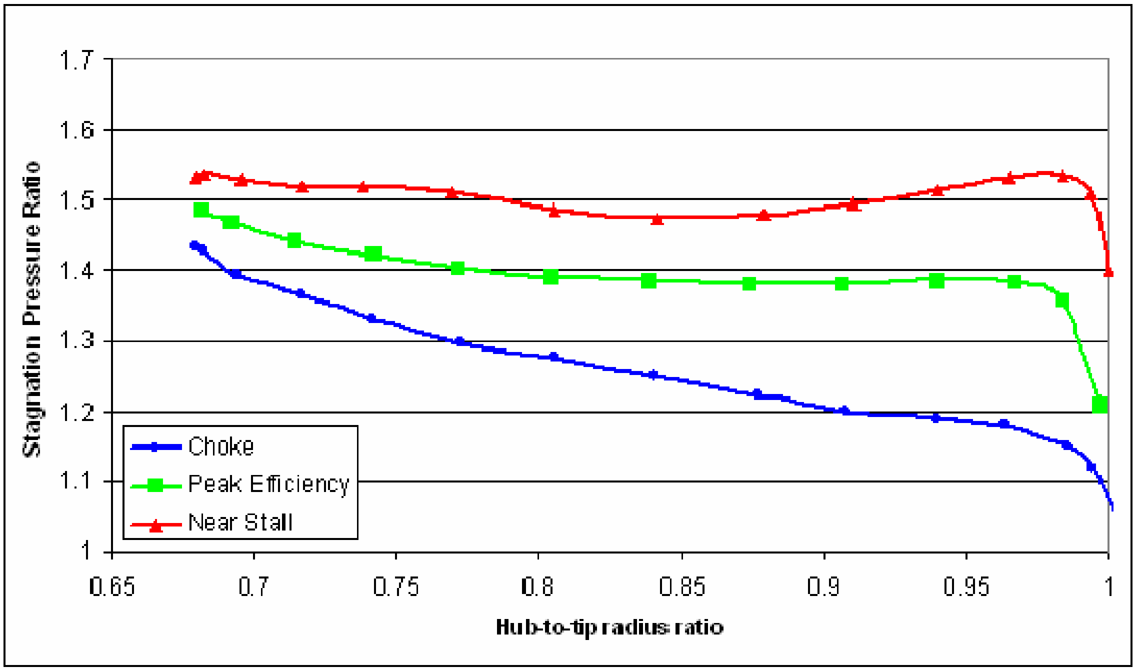
70 % Speed



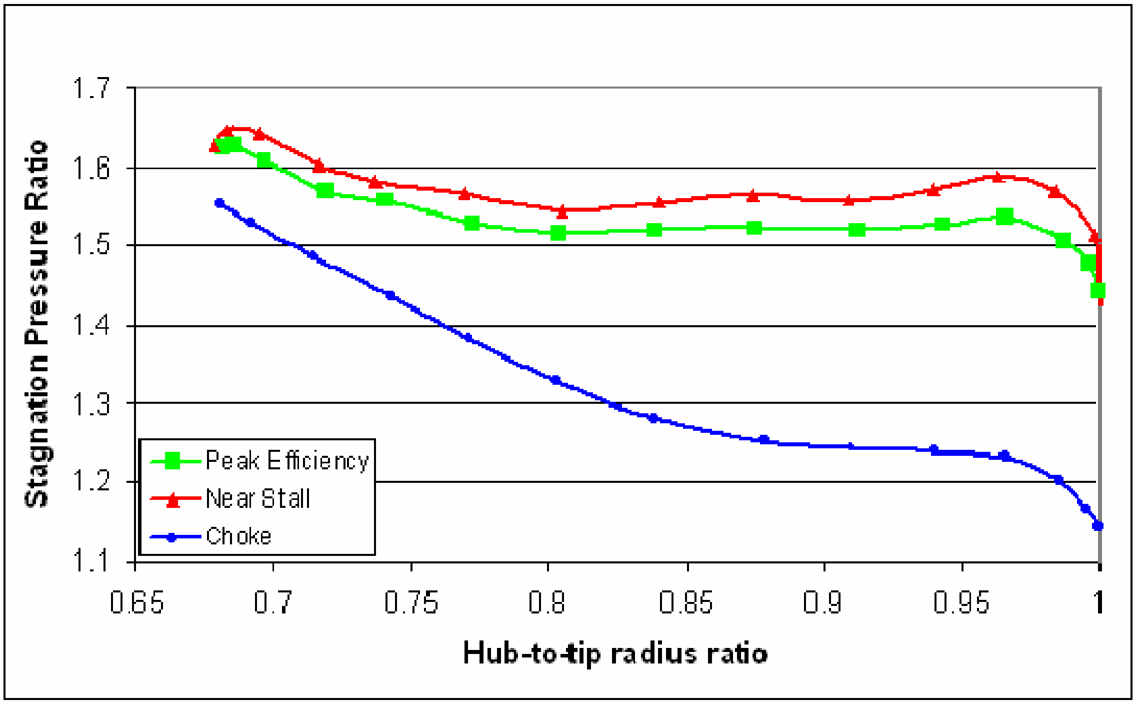
80 % Speed



90 % Speed

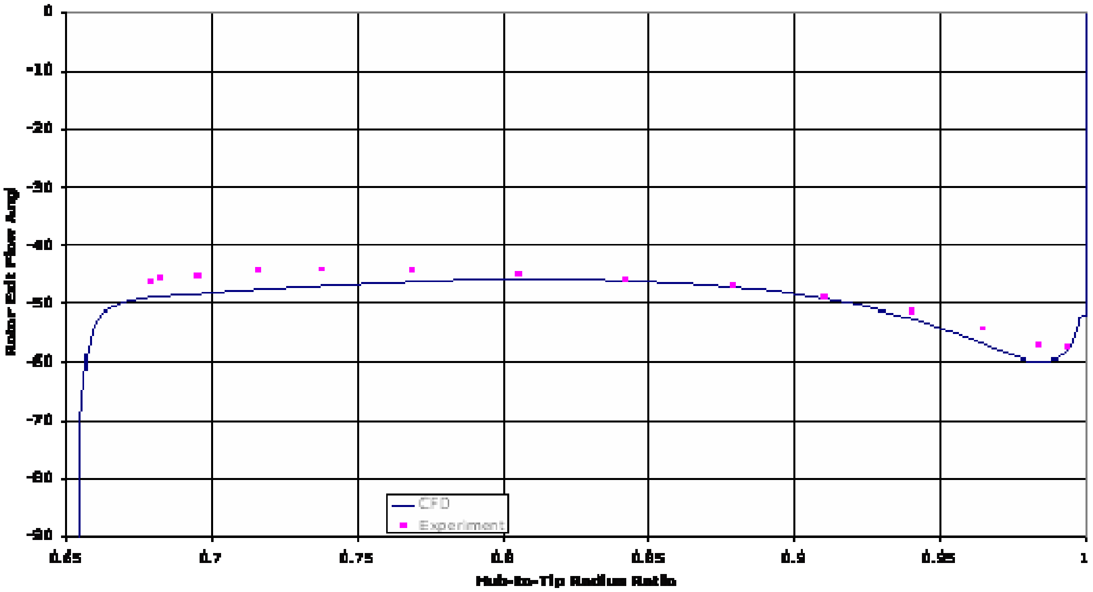


100 % Speed

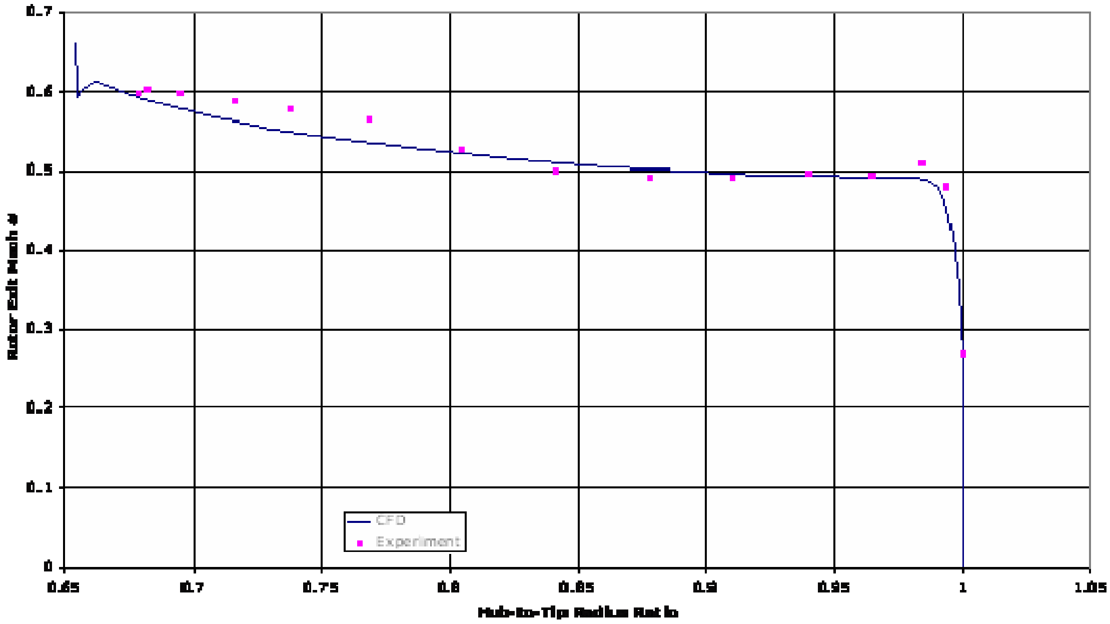


APPENDIX M: CFD VS EXPERIMENTAL DATA: 90% ROTOR SPEED

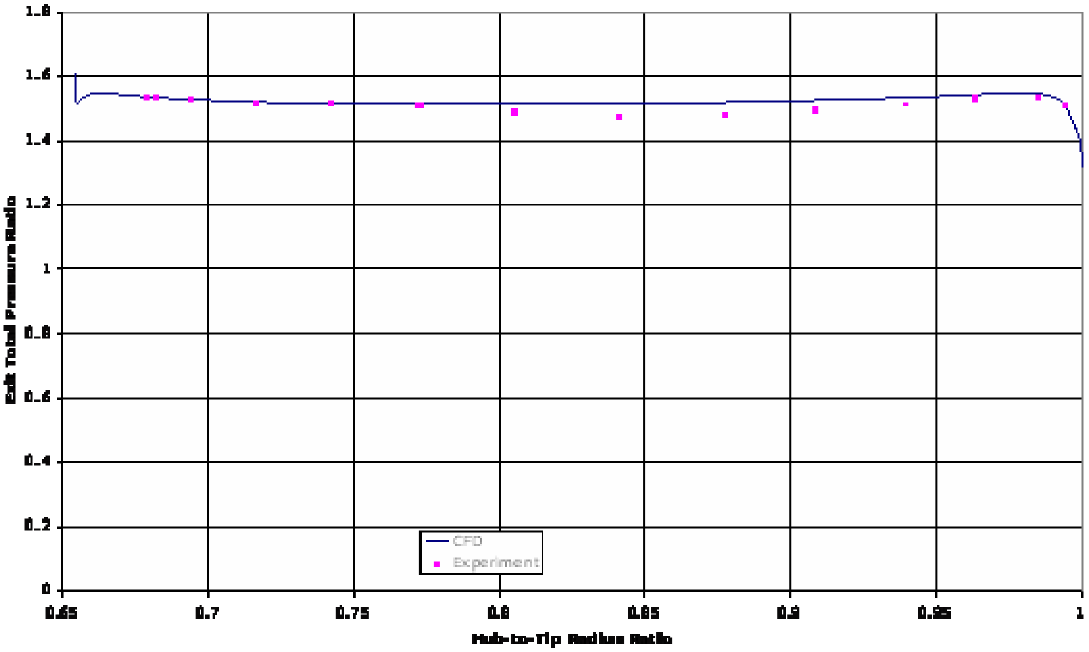
Near Stall Rotor Exit Flow Angle



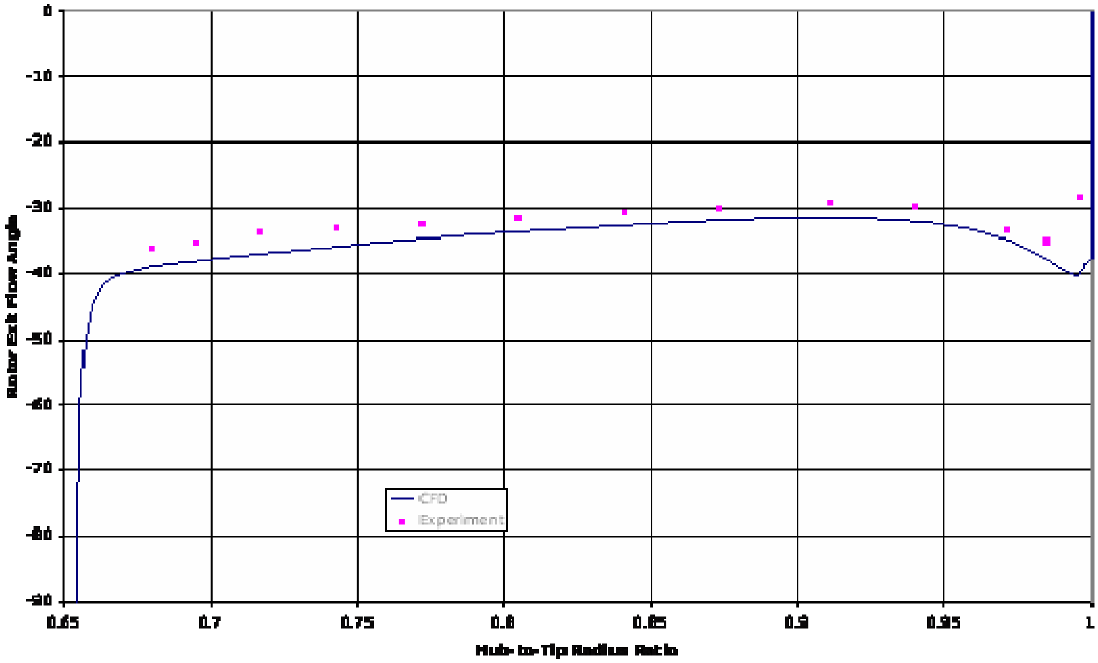
Near Stall Rotor Exit Mach Number



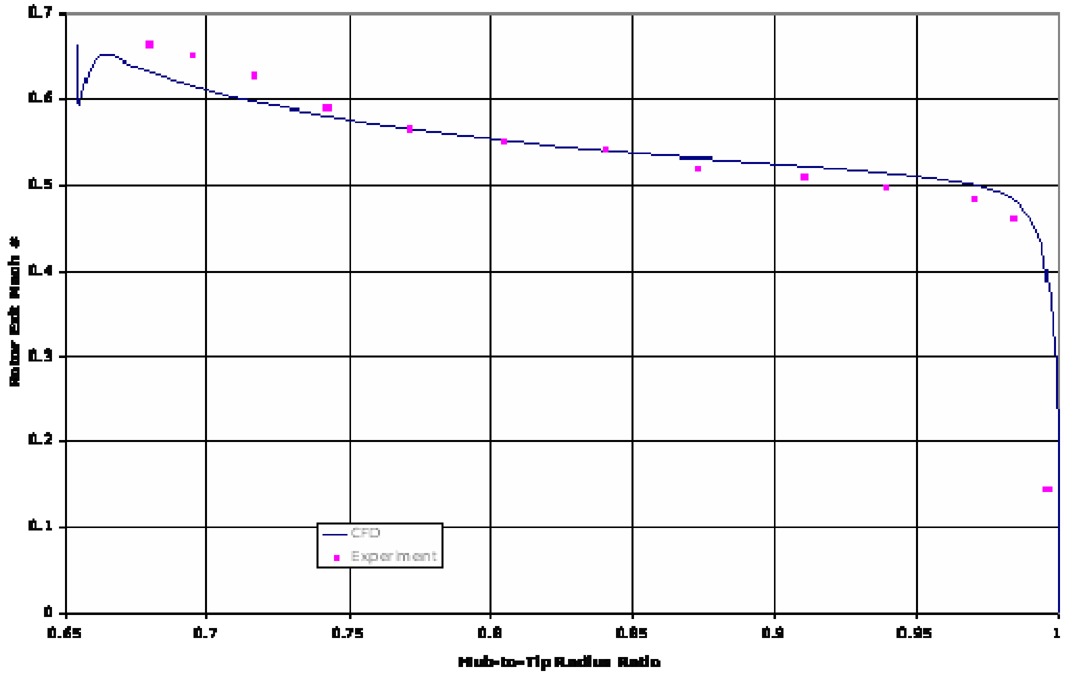
Near Stall Rotor Exit Total Pressure Ratio



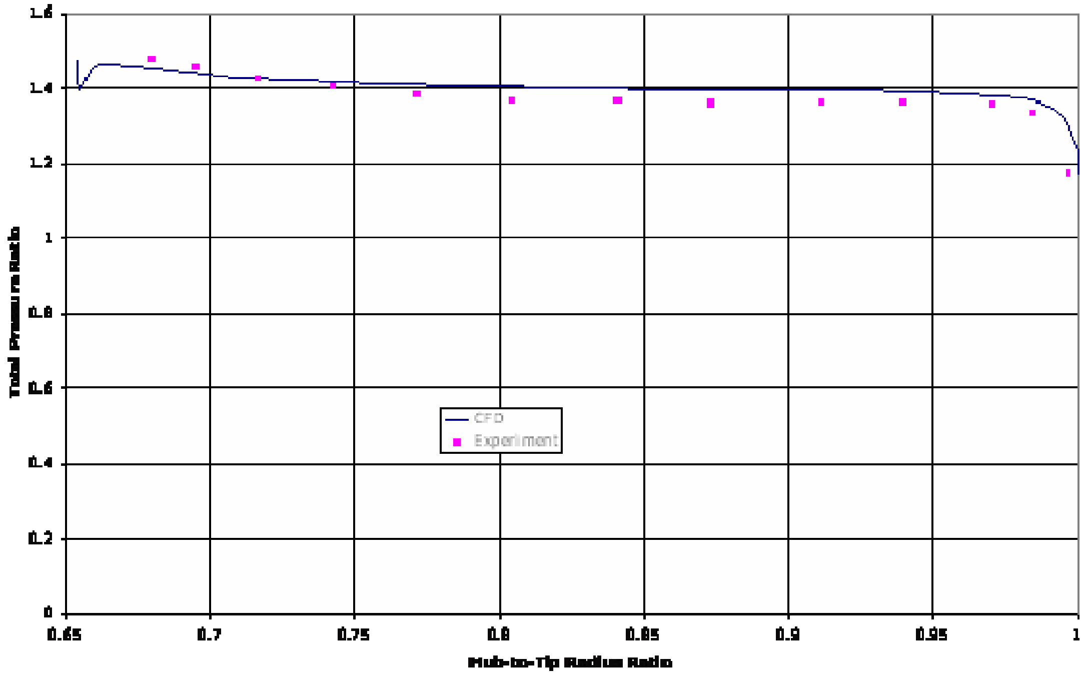
Peak Efficiency Rotor Exit Flow Angle



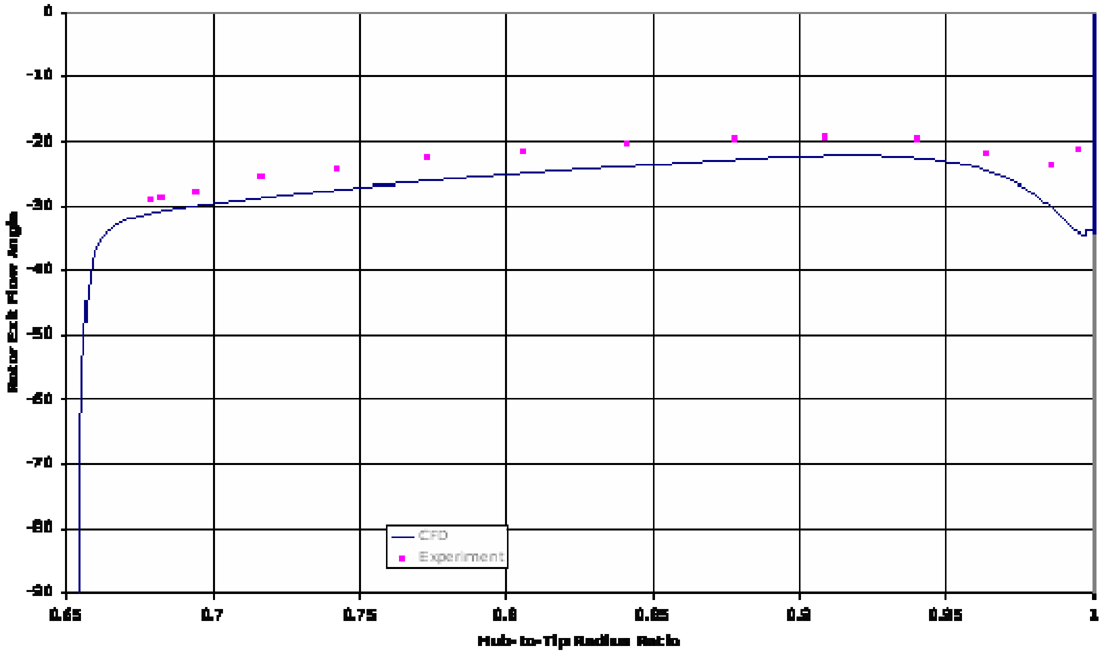
Peak Efficiency Rotor Exit Mach Number



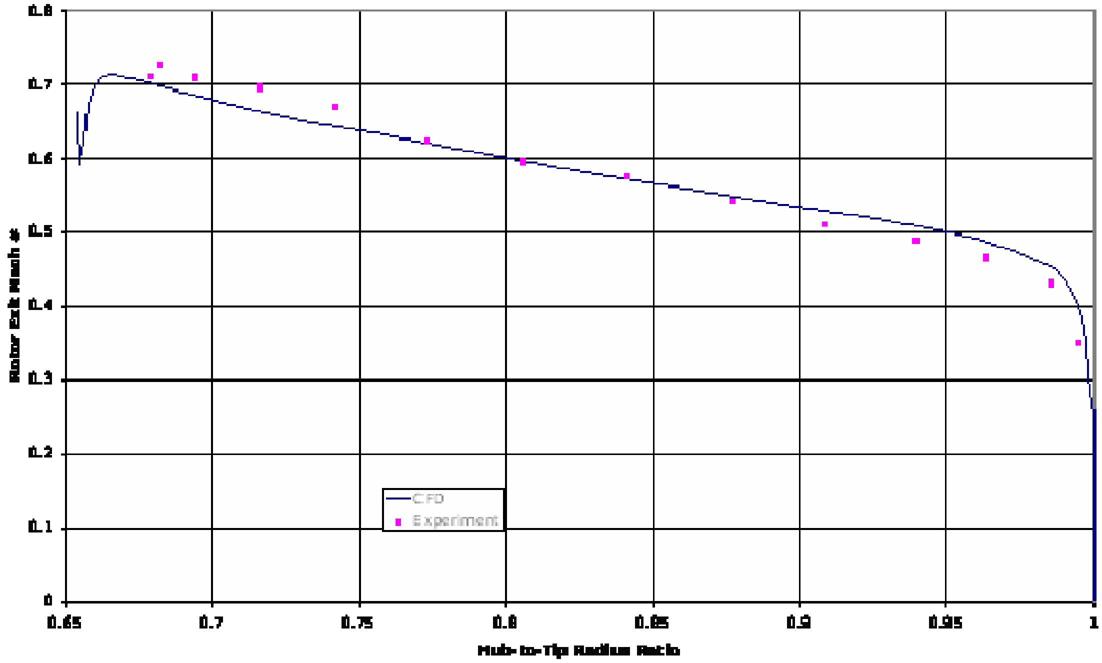
Peak Efficiency Rotor Exit Total Pressure Ratio



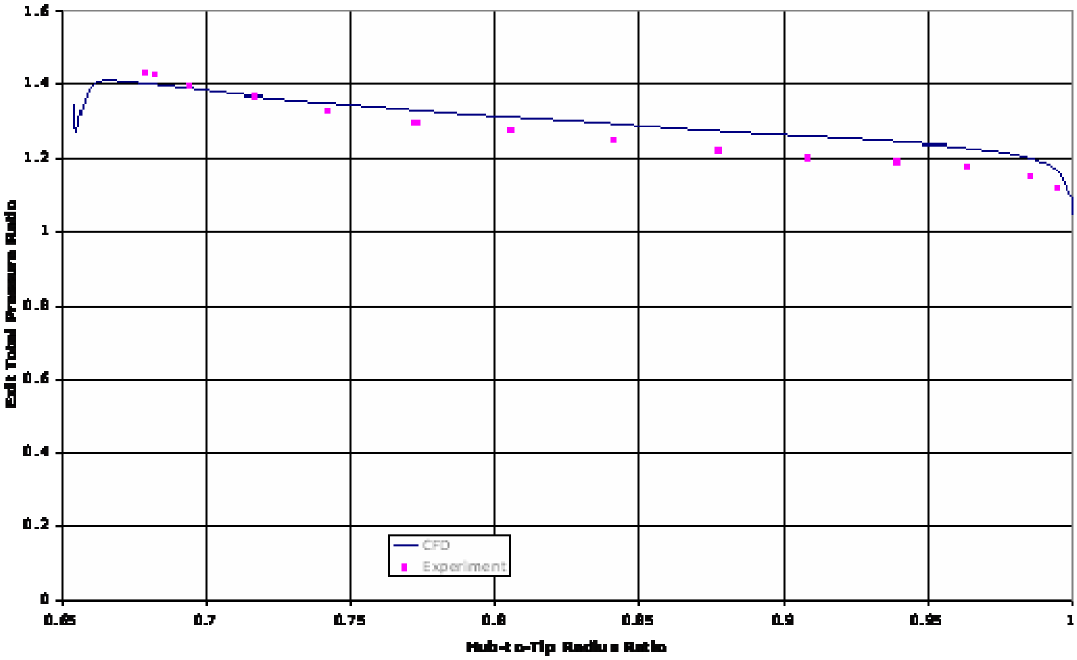
Choke Rotor Exit Flow Angle



Choke Rotor Exit Mach Number



Choke Rotor Exit Total Pressure Ratio



THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX N: MATLAB EXPERIMENTAL DATA COLLECTION FILES

```
function[]=OUTPUT_TO_EXCEL(Tp, Tj)
%this function gets the data from the devices using PMD_1608FS_2 and
%PMD_1608FS_3 and sends the data to ExcelWrite to be written to an Excel
%Spreadsheet.




global Pressures



%Freddy.txt is a notepad value used to hold on to data outside of the
%program to ensure headers will be created.
load freddy.txt


%restart.txt is a notepad value used to hold on to data outside of the
%program to ensure data will not be overwritten if the run is restarted.
load restart.txt


%if it is the first run, or the run has been restarted, this loop will
%place the header in and then write the first line of data.
if freddy==1;


    %if first line, put the header in. if restart, add the number of
    %lines so that data will not be overwritten.
    a=freddy+restart;


    %header
    header={'probe_pos','angle', 'P1', 'P2', 'P3', 'Poj', 'Psj', 'Tp', 'Tj', 'q_probe', 'V_jet',
'Cp1_Probe',...
            'Cp2_Probe', 'Cp3_Probe', 'At_Probe', 'As_Probe',
'Mach_No','Pitch_m','Yaw_m','Mach_m','ps_m','po_m',
'Pitch_m_2','Yaw_m_2','Mach_m_2','ps_m_2','po_m_2','flow_angle'};




    %create a variable that can be evaluated so that the values of lines
    %can be changed.
    fred = ['ExcelWrite(' '''' 'C:\Documents and
Settings\user\Desktop\Transonic\testing1.xls' '''' ', 0,'...
            '''' '[' num2str(a) ',' num2str(2) ']' '''' ', header)'];
```

```
          %evaluate the variable and send the info to ExcelWrite to evaluate it and
          %send it to an EXCEL Spreadsheet.
        eval(fred);



        %change the row number it will be the next available.



        %Call the calculation function
        calculate(Tp, Tj);


        %create a variable that can be evaluated so that the values of lines
        %can be changed.
        %fred = ['ExcelWrite(' '''' 'C:\Documents and Settings\user\Desktop\ERB24\testing1.xls' ''''
', 0,'...
        %       '''' '[' num2str(a) ',' num2str(2) ']' '''' ', m)'];

        %evaluate the variable and send the info to ExcelWrite to evaluate it and
        %send it to an EXCEL Spreadsheet.
        %eval(fred);

        %Change the value of freddy to the next row number.
        %freddy=freddy+1;

        %Save the row number to a text file so that it will not be cleared.
        %save freddy.txt freddy -ascii


        %If the run is not the first, or a reset, just output dats.
        elseif freddy ~=1

          a=freddy+restart;

          %Call the calculation function
          calculate(Tp, Tj);

          %create a variable that can be evaluated so that the values of lines
          %can be changed.
          %fred = ['ExcelWrite(' '''' 'C:\Documents and Settings\user\Desktop\ERB24\testing1.xls'
'''' ', 0,'...
          %       '''' '[' num2str(a) ',' num2str(2) ']' '''' ', m)'];

          %evaluate the variable and send the info to ExcelWrite to evaluate it and
          %send it to an EXCEL Spreadsheet.
          %eval(fred);

          %Change the value of freddy to next row number.
          %freddy=freddy+1;
          %Save the row number to a text file so that it will not be cleared.
          %save freddy.txt freddy -ascii
        end

        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%
        %SUBFUNCTIONS
        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%


        function[header] = calculate(Tp, Tj)
        %this subfunction will get all required data three times and average
        %the three sets of values before outputting the final answer.

        %%%%%%%%%%%%CONSTANTS%%%%%%%%%%%%%%%%%%
        %Cp of air
        Cp_air=1005;

        %gamma of air
        gam_air=1.41;  %gamma of air

        %gravitational constant (m/sec^2)
        g=9.81; %gravitational constant (m/sec^2)

        %density of Mercury (kg/m^3) (calibration pressure in inches of mercury)
        rho_Hg=13550;

        for k=1:2

        %loop for data
        for i=1:3

          %Get the data from the PMD_1608FS's

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%
          %Get the current angle.
          h=PMD_1608FS_2;

          posit=PMD_1608FS_1;
          %Convert the angle to degrees.
          cur_angle(i)=angle_calibration(h);

          probe_pos(i)=traverse_calibration(posit);

          %get the pressure variable info.

          Pressures=pressure_calibration;
          %Pressures=PMD_1608FS_3;
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%
        %Cobra Probe pressures

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%
        %Right Hole Pressure
        P3=Pressures(1);

        %Center Hole Pressure
        P1=Pressures(2);

        %Left Hole Pressure
        P2=Pressures(3);

        %Total Pressure of jet

        Poj=Pressures(4);

        %Static Pressure of jet
        Psj=Pressures(5);

        %%%%%%%%%%%%%%%%%%%%%%
        %Temperature in pipe
        %Tp=is entered manually in the GUI;
        %%%%%%%%%%%%%%%%%%%%%%%%%








        %%%%%%%%%%%%CALCULATIONS%%%%%%%%%%%%%%%%%%%%


        q_probe=P1-(P2+P3)/2;

        V_jet(i)=sqrt(1-(Psj/Poj)^((gam_air-1)/gam_air))*sqrt(2*Cp_air*Tp);


        Cp1_Probe(i)=(P1-P2)/q_probe;

        Cp2_Probe(i)=(P1-P3)/q_probe;

        Cp3_Probe(i)=(P2-P3)/q_probe;
```

```
    At_Probe(i)=(P1-Poj)/q_probe;

    As_Probe(i)=q_probe/(Poj-Psj);

    %Mach Number
    Mach_No(i)=sqrt(((Poj/Psj)^((gam_air-1)/gam_air)-1)*(2/(gam_air-1)));

end



%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Average the values
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

probe_pos=(probe_pos(1) +probe_pos(2)+probe_pos(3))/3;

cur_angle=(cur_angle(1)+cur_angle(2)+cur_angle(3))/3;

Cp1_Probe=(Cp1_Probe(1)+Cp1_Probe(2)+Cp1_Probe(3))/3;

Cp2_Probe=(Cp2_Probe(1)+Cp2_Probe(2)+Cp2_Probe(3))/3;

Cp3_Probe=(Cp3_Probe(1)+Cp3_Probe(2)+Cp3_Probe(3))/3;

V_jet=(V_jet(1)+V_jet(2)+V_jet(3))/3;

At_Probe= (At_Probe(1)+At_Probe(2)+At_Probe(3))/3;

As_Probe=(As_Probe(1)+As_Probe(2)+As_Probe(3))/3;

Mach_No=(Mach_No(1)+Mach_No(2)+Mach_No(3))/3;

Tj=Tj;
m=1;

%Output the data to the GUI
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%
handles = guihandles(gcbo); % generate handles struct



%Set the pressure values to the gui textbox
    set(handles.P_Left, 'String', P3);

    set(handles.P_Center, 'String', P2);

    set(handles.P_Right, 'String', P1);
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%




            %Output and calculate the data from the Three hole probe
            [Pitch_m,Yaw_m,Mach_m,ps_m,po_m] = three_hole_probe_output(m,P3,P2,P1,Poj);

        %Output a second time prelacing Poj with P2 to correct for crazy numbers
        %near the casewall
        [Pitch_m_2,Yaw_m_2,Mach_m_2,ps_m_2,po_m_2]                                    =
three_hole_probe_output(m,P3,P2,P1,P2);



        %flow_angle is the true angle.
        flow_angle=cur_angle+Yaw_m_2-9;



        %Create the data array to be written.
        header=[probe_pos,  cur_angle,  Pressures(3),Pressures(2),Pressures(1),  Pressures(4),
Pressures(5), Tp, Tj, q_probe, V_jet, Cp1_Probe,...
             Cp2_Probe,       Cp3_Probe,       At_Probe,       As_Probe,       Mach_No,
Pitch_m,Yaw_m,Mach_m,ps_m,po_m,        Pitch_m_2,Yaw_m_2,Mach_m_2,ps_m_2,po_m_2,
flow_angle];

        %Set the value of the Mach No textbox to the current Machno.
        set(handles.probemach, 'String', Mach_m_2);







        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%
        %NULL YAW LOOP
        %Added to rotate the probe directly into the flow for increased accuracy
        %when evaluating the flow behind the rotor.


        %flowangle is the angle relative to the probe.
        flowangle=Yaw_m_2;
```

```
    if abs(flowangle)<5 | abs(flowangle)>60;

      break
    elseif flowangle>=0;
        dur=abs(flowangle/2.9);
        ERB24_RIGHT(dur);
        pause(.5);
        current_angle;



      elseif flowangle<=0;
        dur=abs(flowangle/3);
        ERB24_LEFT(dur);
        pause(.5);
        current_angle;

      end
    end
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%



    %Freddy.txt is a notepad value used to hold on to data outside of the
    %program to ensure headers will be created.
    load freddy.txt

    %restart.txt is a notepad value used to hold on to data outside of the
    %program to ensure data will not be overwritten if the run is restarted.
    load restart.txt

    a=freddy+1+restart;

      %create a variable that can be evaluated so that the values of lines
      %can be changed.

      fred          =          ['ExcelWrite('          ''''          'C:\Documents          and
Settings\user\Desktop\Transonic\testing1.xls' '''' ', 0,'...
          '''' '[' num2str(a) ',' num2str(2) ']' '''' ', header)'];

      %evaluate the variable and send the info to ExcelWrite to evaluate it and
      %send it to an EXCEL Spreadsheet.
      eval(fred);

      %pause to ensure Excel is closed for a slow computer
      pause(1)

      %m=1;

      %[Pitch_m,Yaw_m,Mach_m,ps_m,po_m] = three_hole_probe_output(m,P3,P2,P1,Poj)
      %Change the value of freddy to next row number.
      freddy=freddy+1;

      %Save the row number to a text file so that it will not be cleared.
      save freddy.txt freddy -ascii
```

```
function[] = sampledata_run(begpoint, endpoint, points, Tp, Tj)


%numpos is the input number of positions to take a sample of.
%range is the distance to take samples on with zero beginning at the
%casewall

thelength=begpoint-endpoint;

%Create a linear array
thepos=linspace(0, 1, points);


%output an array of positions based on cosine
cospos=abs(1-cos(thepos*pi))/2;

%create the array of final sample positions
sample_pos=cospos*thelength;

%Change the starting area, if needed.
flipped=fliplr(sample_pos);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
final_pos=flipped+endpoint;
%plot(thepos, final_pos)
%break

%for j=1:points
    %handles = guihandles(gcbo); % generate handles struct

    %set(handles.Stop_Points(j), 'String', final_pos(j));
    %end
```

```
for i=1:points

    Pos_Input=final_pos(i);

    handles = guihandles(gcbo); % generate handles struct

    set(handles.Pos_Input, 'String', Pos_Input);

    probe_pos_req=Pos_Input;

    traverse(probe_pos_req);

    pause(1);

    OUTPUT_TO_EXCEL(Tp, Tj);

end
```

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX O: DIFFUSION FACTOR CALCULATIONS

The diffusion factor is a method of assessing the blade loading in an axial compressor. For a simple, two-dimensional geometry, the diffusion factor is calculated using the equation:

$$DF := 1 - \frac{V_2}{V_1} + \frac{V_{\theta 2}}{2 \cdot \sigma \cdot V_1}$$

where V_1 and V_2 are the inlet and exit velocities relative to the blade, $V_{\theta 2}$ is the turning of the flow, assuming the flow is axial at the blade entrance, and σ is the solidity of the blade corresponding to the flow. It is believed that values above 0.6 can indicate blade stall and values of 0.45 may be a typical design choice. (Cumpsty, 1989)

The diffusion factor was calculated for each set of speed data and results were then compared to CFD data created with SWIFT Code. A sample set of calculations for 100% Speed, Peak Efficiency, at a point near the midpoint of the blade is show in the following:

D_s is the design speed in RPM.

$$D_s := 27085$$

ω is the angular velocity in rad/sec.

$$\omega := D_s \cdot 2 \cdot \frac{\pi}{60 \cdot sec}$$

$$\omega = 2.836 \times 10^3 Hz$$

mdot is the mass flow rate in Kg/sec.

$$\mathrm{mdot} := 7.687\,\frac{\mathrm{kg}}{\mathrm{sec}}$$

T_o is the Stagnation Temperature, measured in the upstream flow.

$$T_o := 292\mathrm{K}$$

M is the Mach number, measured by the three hole pressure probe in the upstream flow. M has been area averaged over the 15 data points taken upstream.

$$M := .3344$$

λ is the Specific Heat Ratio. The value of 1.41 is used throughout the calculations for air and was used during the calibration as well.

$$\lambda := 1.41$$

The Static Temperature, T_s is calculated from the Mach number and Stagnation temperature.

$$T_s := \left[\frac{T_o}{1+\left[\frac{(\lambda-1)}{2}\right]\cdot M^2}\right]$$

$$T_s = 285.456\,\mathrm{K}$$

P_s is the Stagnation Pressure, measured with a pitot static tube upstream of the flow.

$$P_s := 69153\text{Pa}$$

R_d is the Ideal Gas Constant specific for air.

$$R_d := 287\cdot\frac{J}{K\cdot kg}$$

The density, ρ is calculated upstream in the flow, prior to decrease in area caused by contraction in area prior to entering the turbine blades.

$$\rho := \frac{P_s}{R_d\cdot T_s}$$

$$\rho = 0.844\frac{kg}{m^3}$$

The diameter of the inlet, d, is 0.2794 meters (11inches).

$$d := .2794\cdot m$$

A_t is the total area of the inlet.

$$A_t := \pi \cdot \frac{d^2}{4}$$

$$A_t = 0.061312\,m^2$$

Knowing the mass flowrate, density, and area, allows us to calculate $V_{upstream}$, the velocity of the fluid prior to reaching the spinner.

$$V_{upstream} := \frac{mdot}{\rho \cdot A_t}$$

$$V_{upstream} = 148.534\,\frac{m}{s}$$

The area of the inlet is reduced as the air is directed through the compressor blades. The spinner and reduction in area acts as a nozzle, and a new velocity must be calculated to better approximate the velocity at the leading edge of the blades. The restriction in area is treated as a simple nozzle in the following equations.

$$A_{entrance} := A_t - \frac{\pi \cdot (2 \cdot .073025 \cdot m)^2}{4}$$

$$A_{entrance} = 0.044559\,m^2$$

The critical area A_{star} can be calculated assuming isentropic flow and that there are no shock waves in the duct. (White, 2003)

$$A_{star} := \frac{\left(A_t \cdot M\right)}{\left[\frac{\left[1 + .5\cdot(\lambda - 1)\cdot M^2\right]}{.5\cdot(\lambda + 1)}\right]^{.5\cdot(\lambda+1)\cdot(\lambda-1)}}$$

$$A_{star} = 0.022231\,\mathrm{m}^2$$

Once the critical area is know, the same equation is then used to solve for the Mach number at the entrance to the compressor blades. The sixth order equation was solved using Matlab.

$$A_{\ star} := \frac{\left(A_{\ t} \cdot Ma\right)}{\left[\frac{\left[1 + .5\cdot(\lambda - 1)\cdot Ma^2\right]}{.5\cdot(\lambda + 1)}\right]^{.5\cdot(\lambda+1)\cdot(\lambda-1)}}$$

Of the six solutions, only two were non-imaginary. The first could not be used as it was greater than 1 and the second satisfied the requirements of being less than one:

$$Ma := 0.461935$$

The Mach number was then designated M_1, the mach number of the flow directly before the leading edge of the compressor blades.

$$M_1 := .461935$$

The density of the entrance flow could then be calculated.

$$\rho_e := \rho\left[1 + .5\cdot(\lambda - 1)\cdot M_1^{\,2}\right]^{\left[\frac{1}{(\lambda-1)}\right]}$$

$$\rho_e = 0.937\,\frac{kg}{m^3}$$

The Static Temperature at the entrance T_{se} could then be calculated, assuming isentropic conditions, as the Stagnation Temperature T_o remains constant in a nozzle.

$$T_{se} := \left[\frac{T_o}{1 + \left[\frac{(\lambda - 1)}{2}\right]\cdot M_1^{\,2}}\right]$$

$$T_{se} = 279.762\,K$$

With the Mach number and Static Temperature, the Static Velocity C_1 could then be calculated.

$$C_1 := M_1 \cdot \sqrt{\lambda \cdot R_d \cdot T_{se}}$$

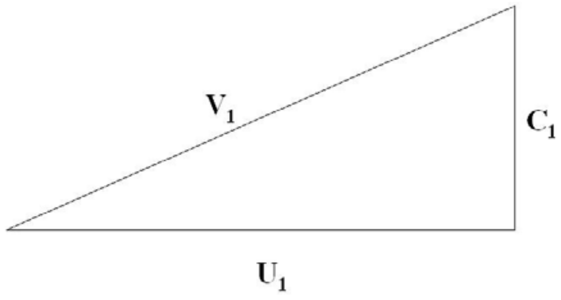
C_1 is then the velocity of the flow as it enters the blade row.

$$C_1 = 155.427 \frac{m}{s}$$

The radial distance, r_e, at the leading edge of the blade was used for calculating the relative velocity of the blade at the flow entrance. The distance was calculated matching the flow area where the data point was taken downstream, to the flow area at the trailing edge of the blade. This flow area was then matched to streamlines calculated by the CFD Swift code.

$$r_e := .11925m$$

Entrance Flow Velocity Triangle



U_1 is the velocity at the leading edge of the blade at the point that is measured. The points that are used for U_1 are the area averaged points corresponding to the area averaged points sampled. It is a product of the rotational speed and radial distance from the center.

$$U_1 := \omega \cdot r_e$$

$$U_1 = 338.233 \frac{m}{s}$$

V_1 is the inlet velocity relative to the blade.

$$V_1 := \left(\sqrt{{C_1}^2 + {U_1}^2} \right)$$

$$V_1 = 372.235 \frac{m}{s}$$

$V_{\theta 1}$ was assumed to be zero, as the flow was found to be axial upstream of the compressor.

The flow variables were then calculated downstream of the blade row. The flow was sampled at 15 positions using the pressure probe.

M_2 is the Mach number measured of the flow measured with the three-hole pressure probe at point four downstream in the case wall. The position was taken near the midpoint of the flow.

$$M_2 := .555$$

The Mach number allows us to calculate the static temperature, assuming the Stagnation Temperature has remained constant through the compressor.

$$T_{s2} := \left[\frac{T_o}{1 + \left[\frac{(\lambda - 1)}{2} \right] \cdot M_2{}^2} \right]$$

$$T_{s2} = 274.657\,K$$

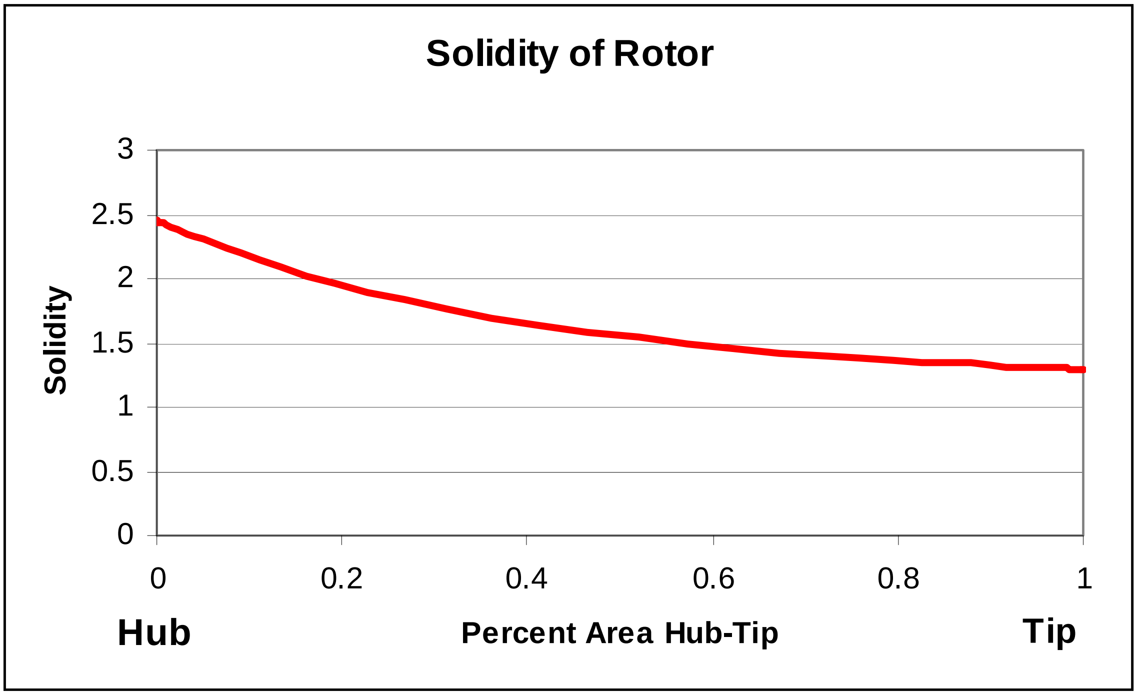
The Static Temperature allows us to calculate the Static Velocity of the flow downstream of the compressor rotor.

$$C_2 := M_2 \cdot \left(\sqrt{R_d \cdot \lambda \cdot T_{s2}} \right)$$

$$C_2 = 185.029\,\frac{m}{s}$$

The solidity, σ was matched to the area integral of flow of the trailing edge of the blade. The solidity was calculated using flow lines taken at 49 points from a CFD grid along the actual blade dimensions following the flow line.

The solidity data was graphed and is shown below. Solidity calculated matched the predicted values by Sanger. (Sanger, 1996)



The solidity was then interpolated for the points required for each experimental data point based on the ratio of the area of flow.

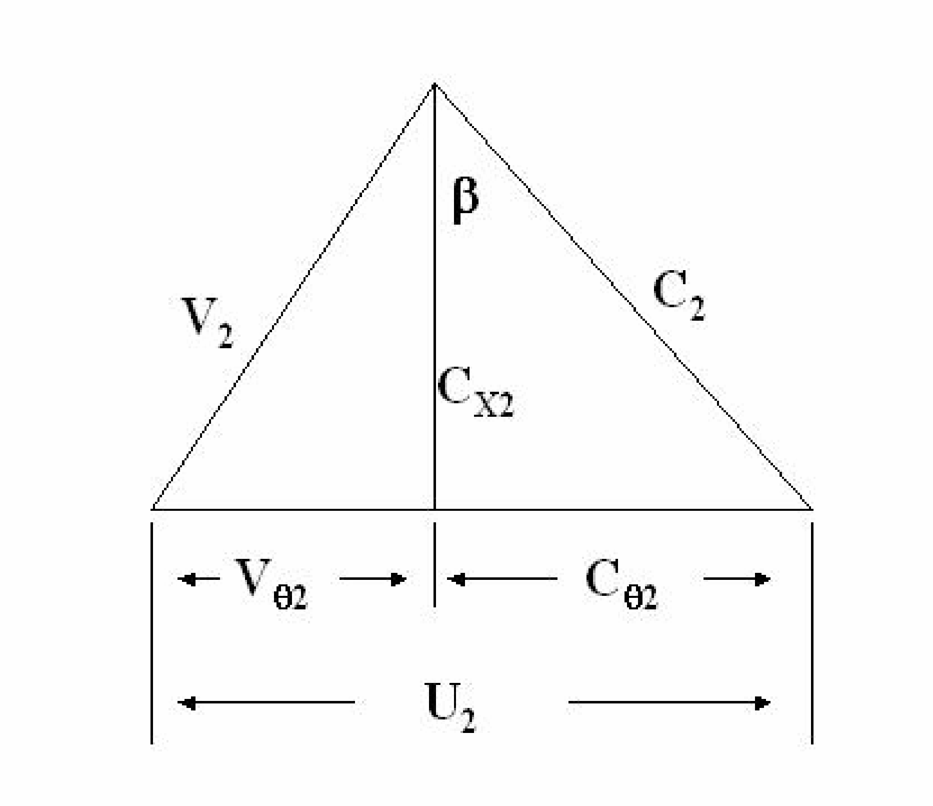
For this data point solidity was found to be:

$$\sigma := 1.642$$

The value for β, the yaw angle of the flow was provided by the three-hole probe from the values taken downstream of the rotor. The angle was used to calculate the velocity relative to the blade. The velocity triangle is shown below in Figure 45.

$$\beta := 25.27 \cdot \mathrm{deg}$$

Exit Flow Velocity Triangle



The variable $C_{\theta 2}$ is the axial component of the static flow velocity and is calculated using the static velocity and the angle of the flow.

$$C_{\theta 2} := C_2 \cdot \sin(\beta)$$

$$C_{\theta 2} = 78.986 \frac{m}{s}$$

C_{X2} is the radial component of the static flow velocity and is also calculated with the static velocity and angle of the flow found by the probe.

$$C_{X2} := C_2 \cdot \cos(\beta)$$

$$C_{X2} = 167.322 \frac{m}{s}$$

U_2 was the velocity at the trailing edge of the blade at the area averaged point of measurement. Like U_1 it was a product of the rotational speed and the distance from the center of the measurement point, however, in this case the point was projected onto the trailing edge of the rotor blades. The radial velocity is found by measuring the distance the point was from the case wall, subtracting it from the full radius, and multiplying the result by the angular velocity.

$$U_2 := (0.1397 - 0.0228223)m \cdot \omega$$

$$U_2 = 331.504 \frac{m}{s}$$

$V_{\theta 2}$ was the angular relative velocity and was the difference between the velocity of the trailing edge and the axial component of flow.

$$V_{\theta 2} := U_2 - C_{\theta 2}$$

$$V_{\theta 2} = 252.518 \frac{m}{s}$$

V_2 was the velocity of the flow, relative to the blade. With the angular and axial flow components found, V_2 was calculated.

$$V_2 := \sqrt{C_{X2}^{\;2} + V_{\theta 2}^{\;2}}$$

$$V_2 = 302.923\,\frac{m}{s}$$

With all components of the equation, the Diffusion Factor can then be determined with the diffusion factor equation:

$$DF := 1 - \frac{V_2}{V_1} + \frac{V_{\theta 2}}{2\cdot\sigma\cdot V_1}$$

$$DF = 0.393$$

THIS PAGE INTENTIONALLY LEFT BLANK

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California

3. Distinguished Professor and Chairman Anthony Healey
   Department of Mechanical and Aeronautical Engineering
   Naval Postgraduate School
   Monterey, California

4. Professor Ray Shreeve
   Department of Mechanical and Aeronautical Engineering
   Naval Postgraduate School
   Monterey, California

5. Professor Garth Hobson
   Department of Mechanical and Aeronautical Engineering
   Naval Postgraduate School
   Monterey, California

6. Dr. Anthony Gannon
   Department of Mechanical and Aeronautical Engineering
   Naval Postgraduate School
   Monterey, California

7. Naval Air Warfare Center
   Propulsion and Power Engineering
   ATTN: Mark Klein
   Patuxent River, Maryland

8. LT Ivan Villescas
   Monterey, California